Clarity Insights Data Science Challenge:

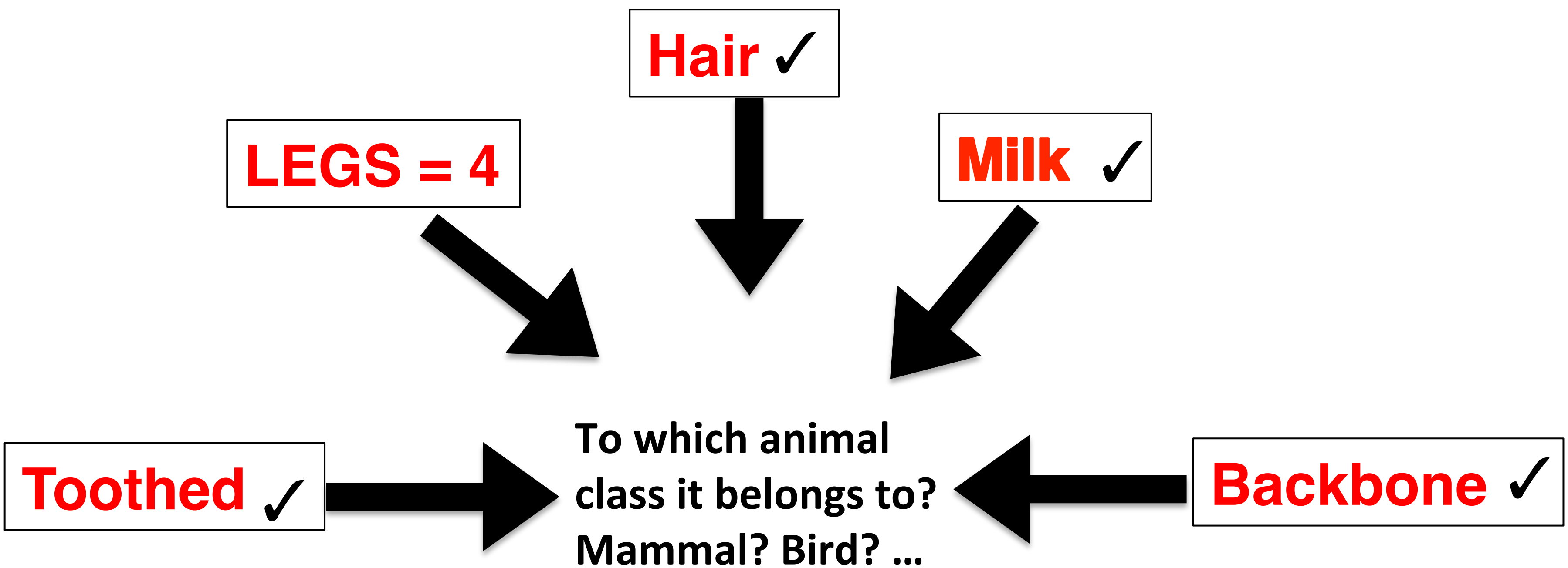
Zoo Animal Classification

Jaime A. Millan

11/09/17

Challenge Description

Given an the these traits, predict to which animal class type this animal belongs to:



Task : Develop a classification algorithm

Dataset Overview

- Dataset consists of 101 animals from a zoo.
- There are 16 features (hair, feathers, eggs, milk, airborne, aquatic, predator, toothed, backboned, breathes, venomous, fins, legs, tail, domestic, catsize).
- There are 7 animal class types: Mammal, Bird, Reptile, Fish, Amphibian, Bug and Invertebrate.
- The task is to be able to predict the classification animals with the aid of a machine learning algorithm.

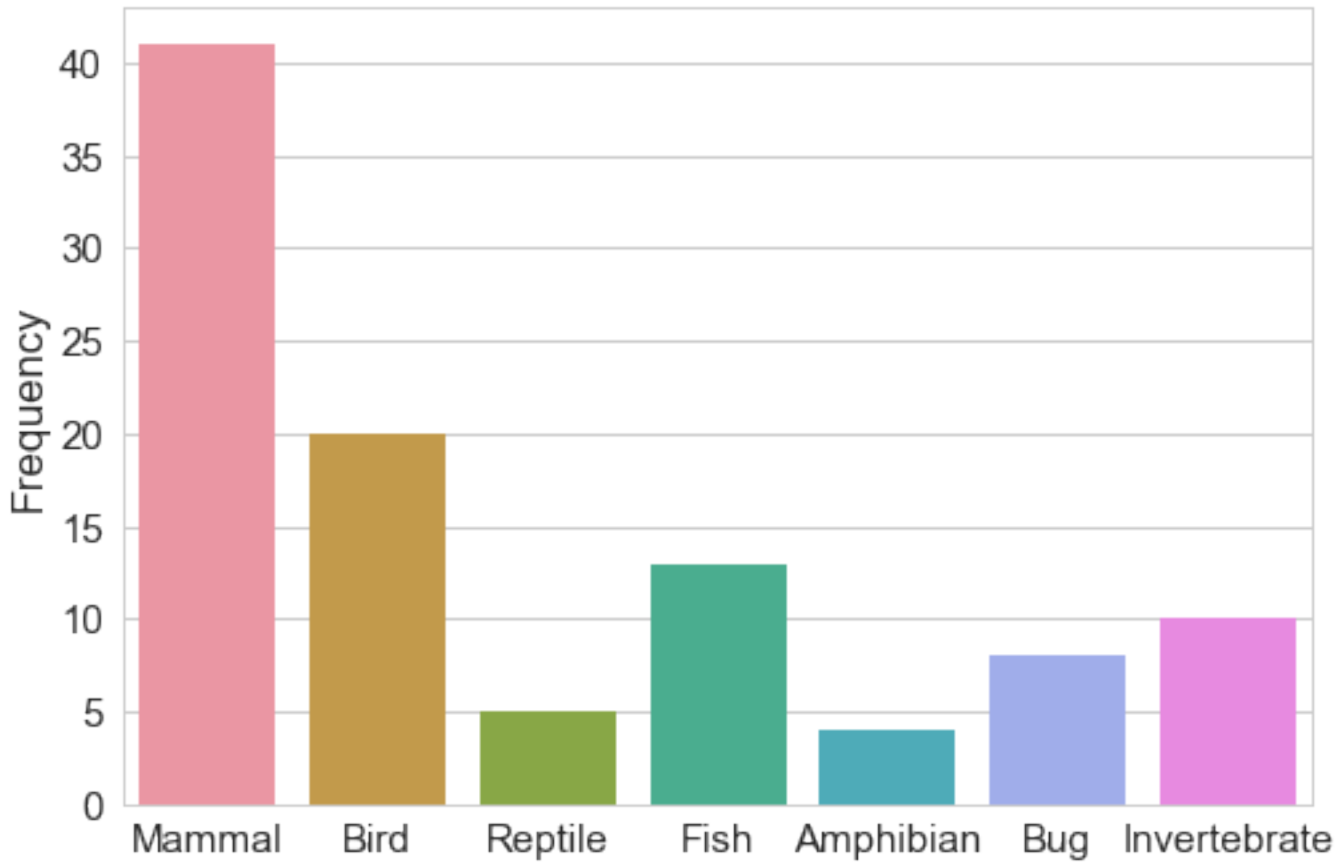
Data Cleaning/Description

- Dataset is quite clean (No entries missing!)
- Al features are boolean
- Analysis perfomed (info() and describe())

```
In [280]:  data. info()
<class
'pandas.core.frame.DataFrame'>
Int64Index: 101 entries, 0 to 100
Data columns (total 18 columns):
animal name 101 non-null object
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
memory usage: 15.0+ KB
0.1
```

```
In [281]:  data. describe()
Out[281]:  hair feathers eggs milk airborne aquatic n
 count 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
std 0.496921 0.400495 0.495325 0.493522 0.427750 0.481335
min 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
50% 0.000000 0.000000 1.000000 0.000000 0.000000 0.000000
75% 1.000000 0.000000 1.000000 1.000000 0.000000 1.000000
max 1.000000 1.000000 1.000000 1.000000 1.000000 1.000000
```

Animal Class Distribution

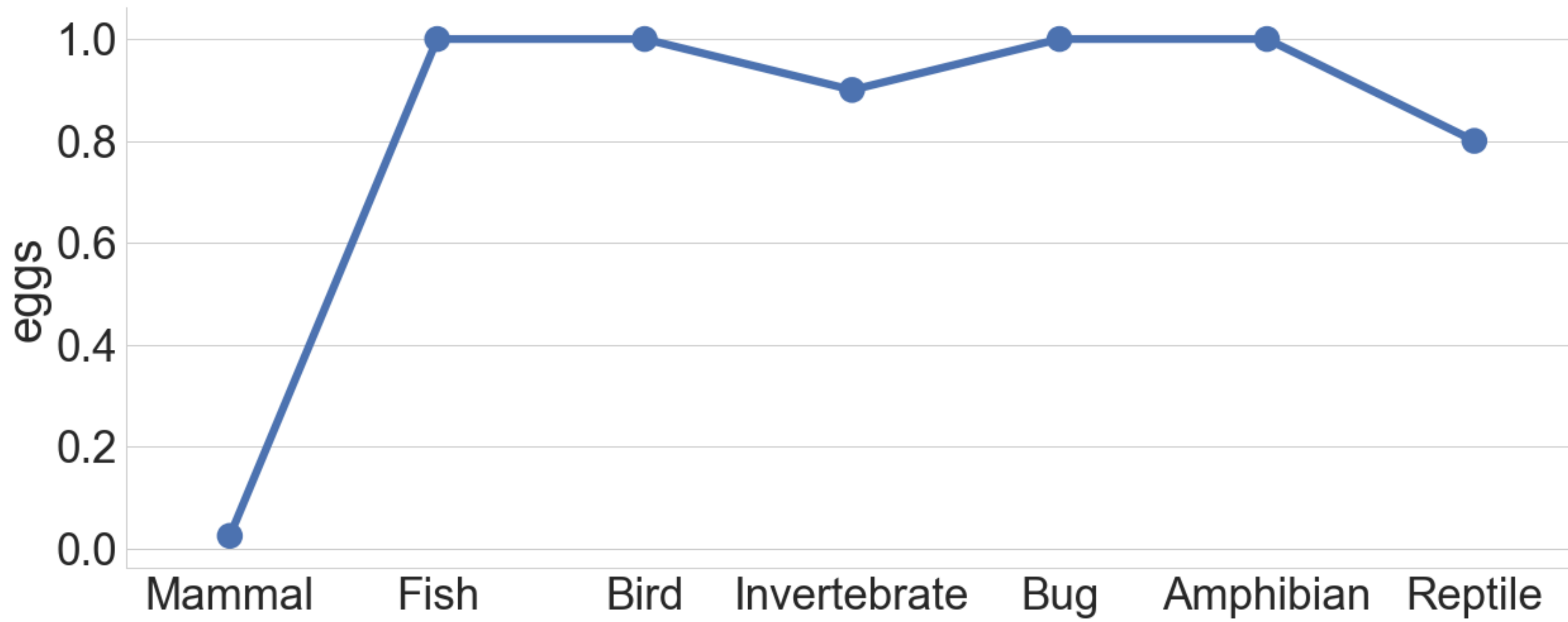


* Sample is small and some animal classes might be undersampled,

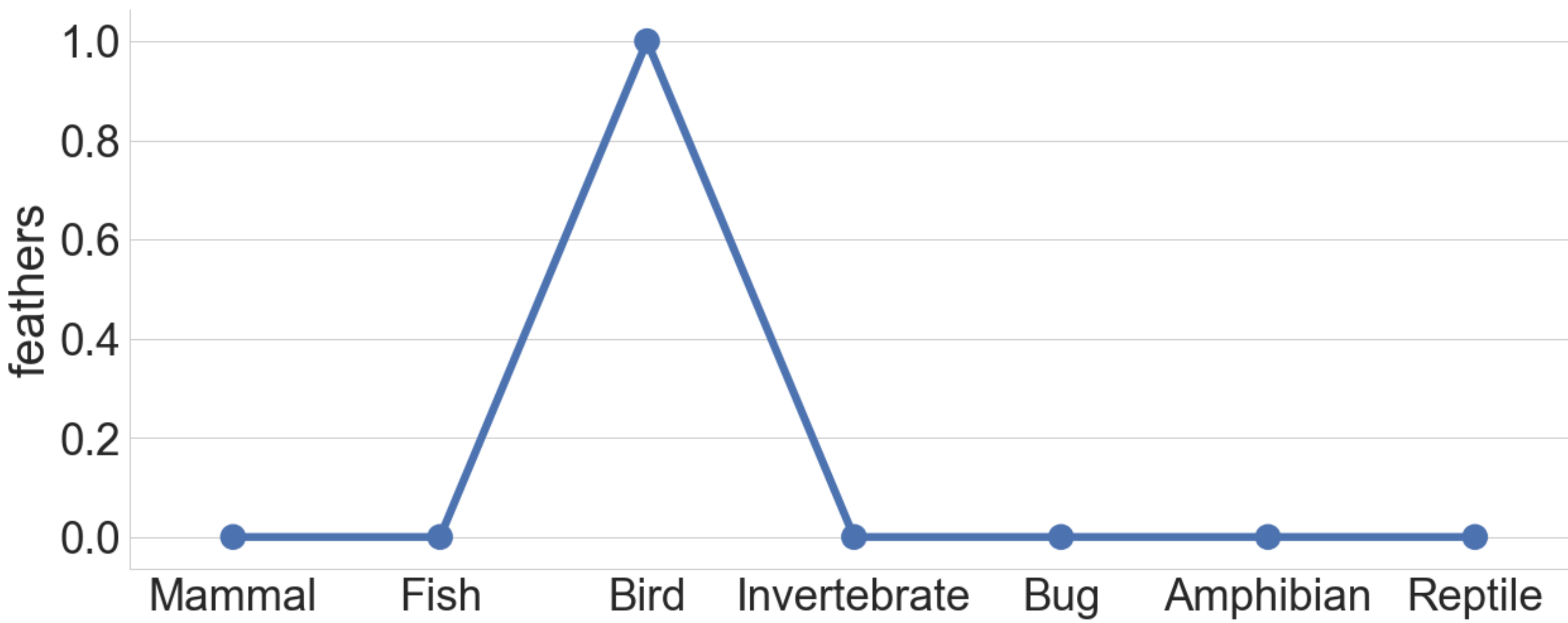
thus so careful model selection/tuning and cross validation is needed

Feathers Mean Value Distributions

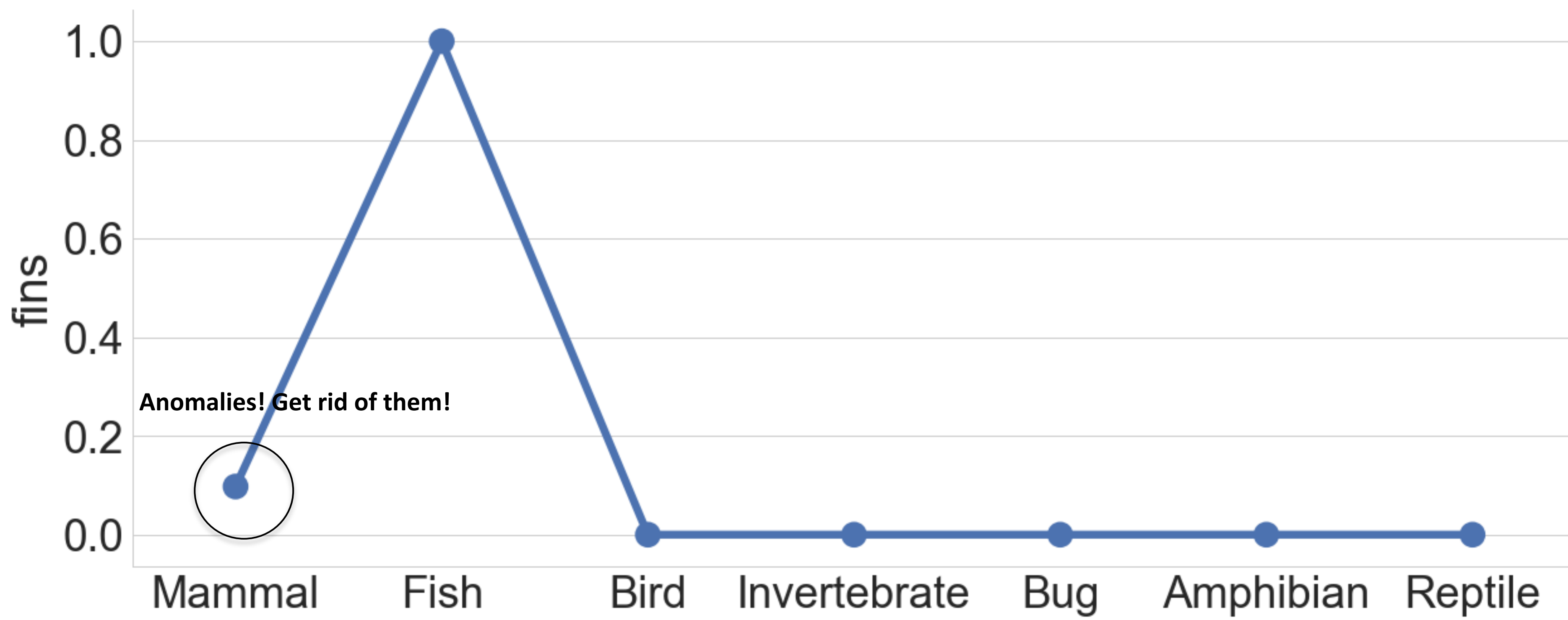
```
In [295]: #Countplot of species depending of features
          sns.set_context("notebook", font_scale=3.5, rc={"lines.linewidth": 3.5})
```



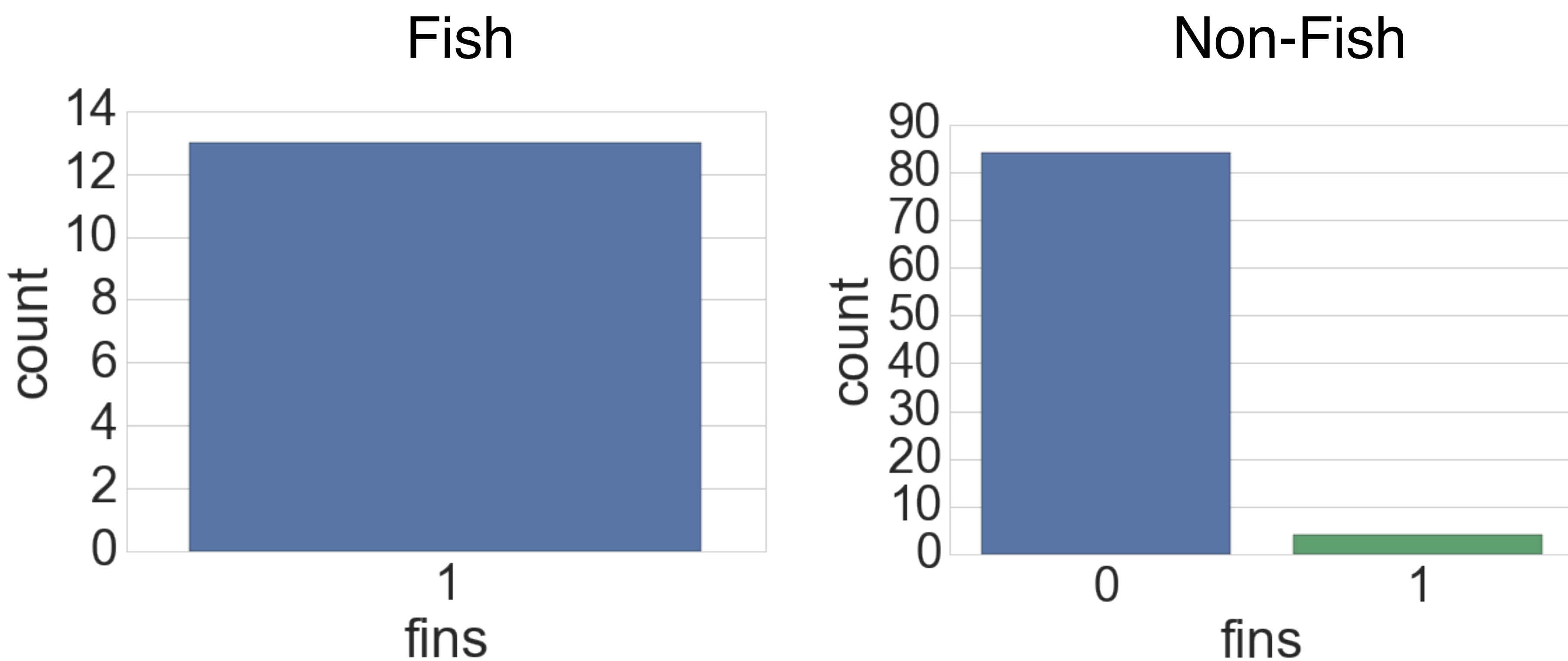
‘Feathers’ Feature Mean Value Distributions



Fins Feature Mean Value Distributions

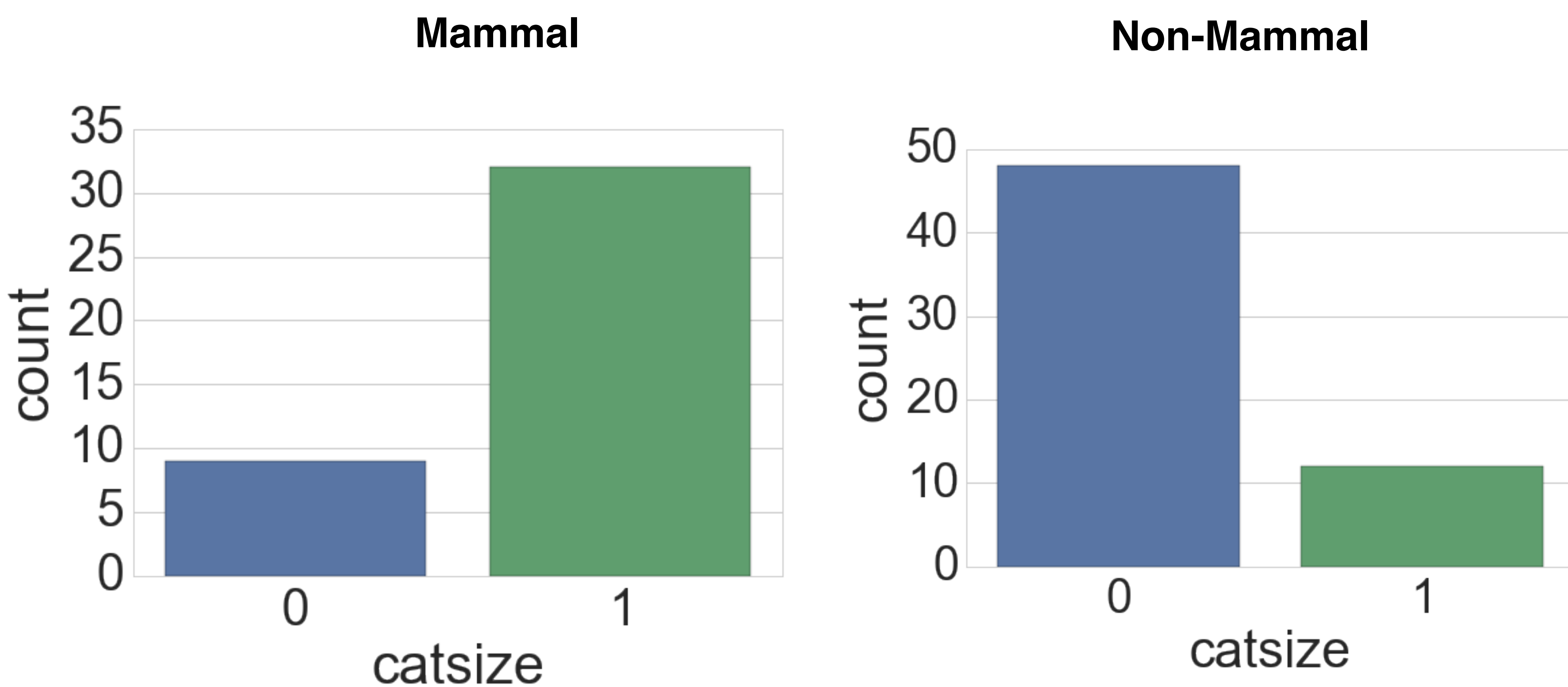


Importance of Fins trait



*This feature could be importance in one-vs-Rest (p_value =4.86e-15)

Importance of “Catsize” trait

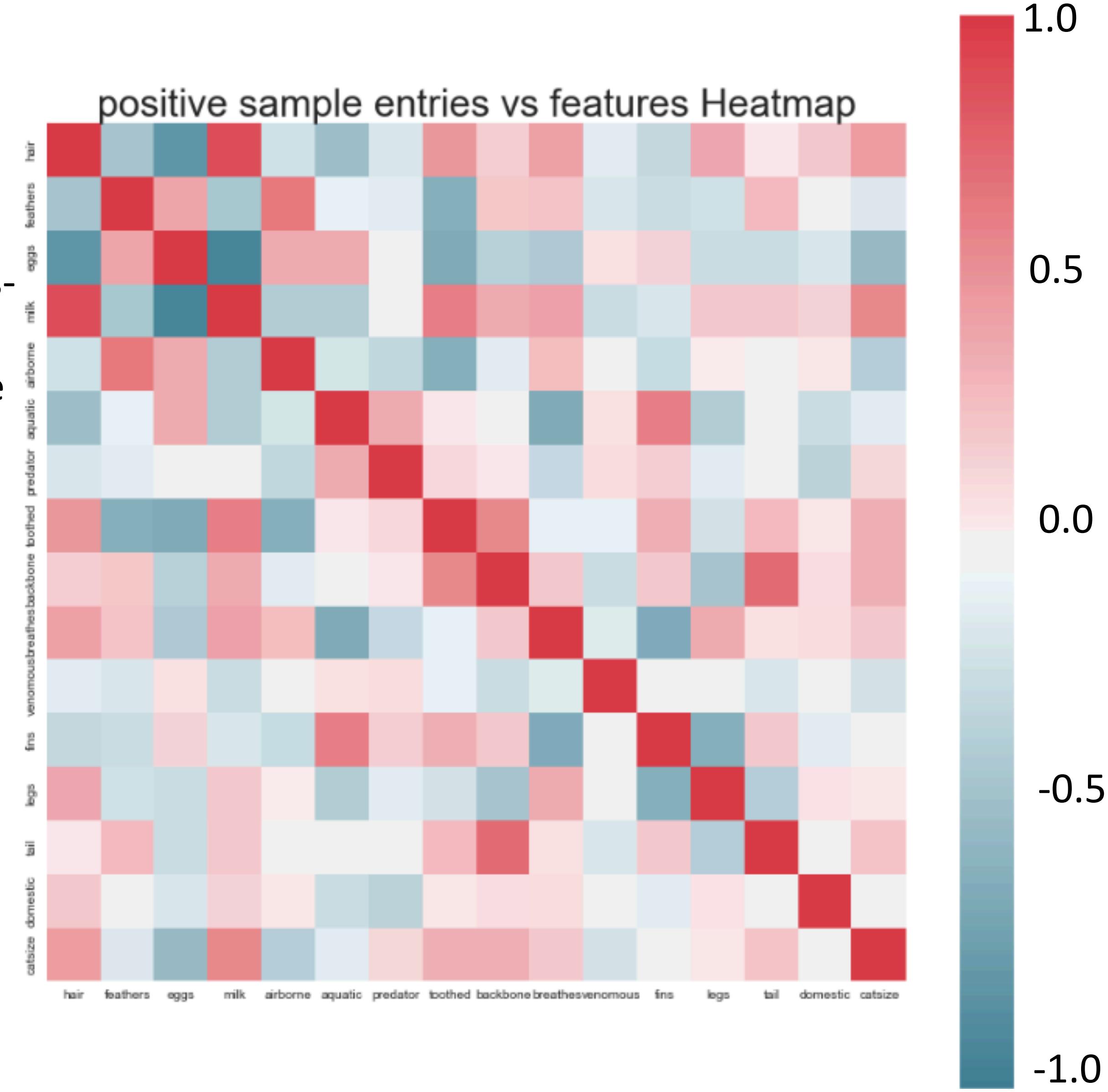


*This feature could be importance in one-vs-Rest (p_value =1.49e-5)

Feature Pearson Correlation

*Eggs-hair and Eggs-milk anti-correlate
*Milk-hair correlate

(P-values ~0)



Feature Importance: Chi-Square Test

Animal Class	Feature A = 1	Feature A = 0	
Mammal	O_i=80 (E_i = 50)	O_i=20 (E_i= 50)	100
Bird	O_i= 20 (E_i= 50)	O_i= 80 (E_i= 50)	100
Total	100	100	200

$$\chi^2 = \sum_{i=1}^{n} \frac{(O_i - E_i)^2}{E_i}$$

= $(80-50)^2$ / 50 + …. = 72 , p-value =1.59e-15 (wikipedia.org)

In [19]:
```
#Select Features based on Values
from sklearn.feature_selection import SelectKBest
from sklearn.feature_selection import chi2

Chi2Selector=SelectKBest(chi2, k=15)
scores,pvalues = Chi2Selector.score_func(X_train,Y_train)
```

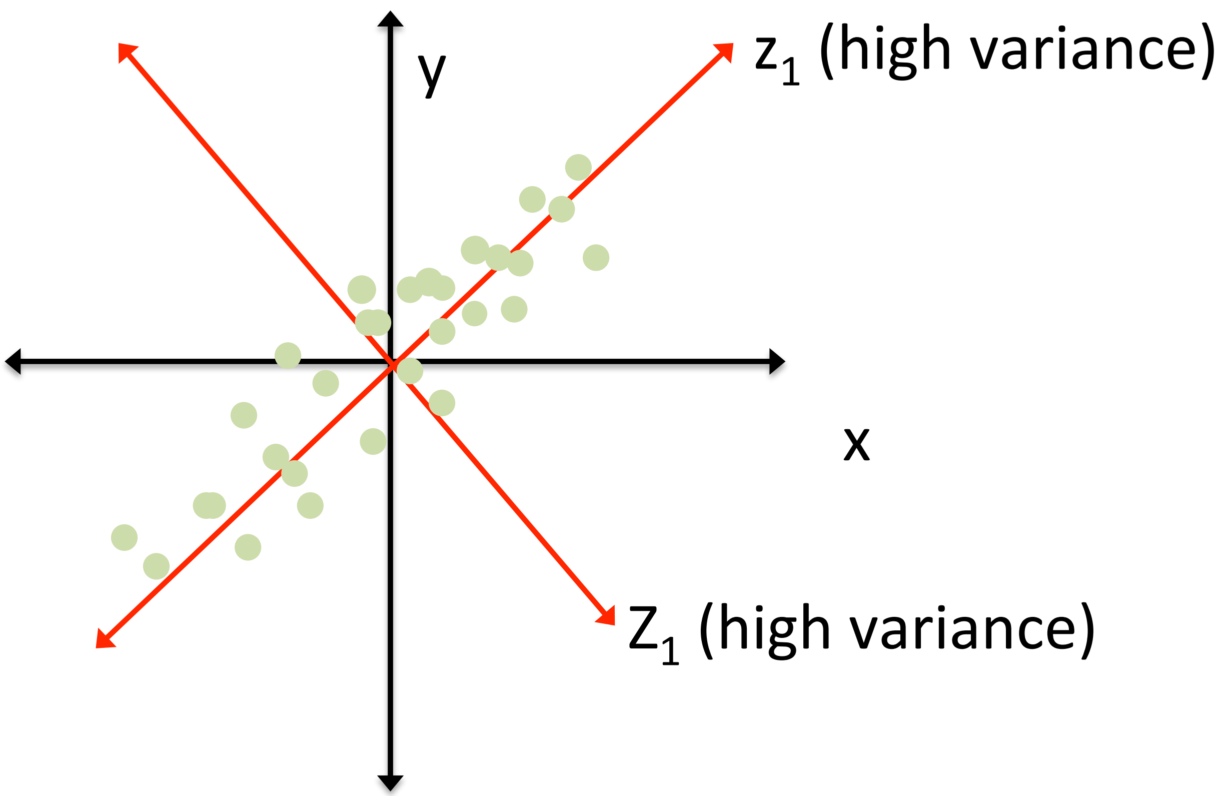
In [21]:
```
coeff_df = pd.DataFrame(X_train.columns)

coeff_df.columns = ['Feature']

coeff_df["Scores"] = pd.Series(scores)
coeff_df["Pvalues"] = pd.Series(pvalues)

coeff_df.sort_values(by='Scores', ascending=False)
```

Out[21]:

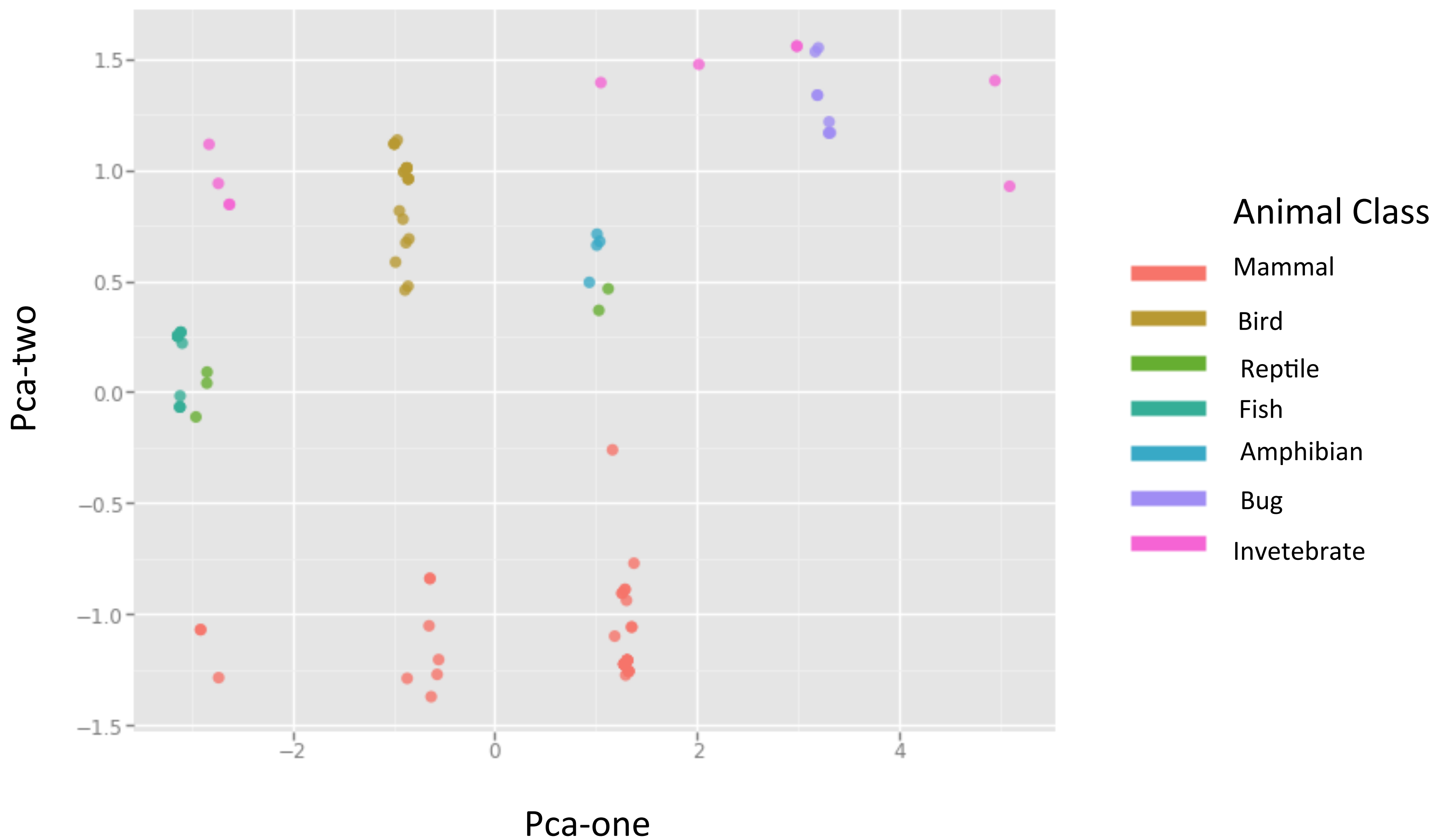
	Feature	Scores	Pvalues
12	legs	81.170621	2.047031e-15
1	feathers	81.000000	2.220198e-15
11	fins	62.553802	1.360770e-11
3	milk	60.000000	4.501017e-11
4	airborne	49.214736	6.753324e-09
0	hair	48.833806	8.049349e-09
2	eggs	37.419719	1.458360e-06
7	toothed	37.060056	1.714052e-06
5	aquatic	29.868970	4.162830e-05
15	catsize	20.976006	1.852915e-03
8	backbone	18.000000	6.232195e-03
10	venomous	17.589904	7.343089e-03
9	breathes	17.338750	8.115690e-03
13	tail	16.880439	9.732839e-03
6	predator	5.430810	4.898584e-01
14	domestic	4.192521	6.506412e-01

Principal Component Analysis (PCA)

- PCA finds orthorgonal dimensions (eigenvectors of covariance matrix)
- It ranks this dimensions by its variance (eigenvalues of covariance matrix)



PCA 2D Plots



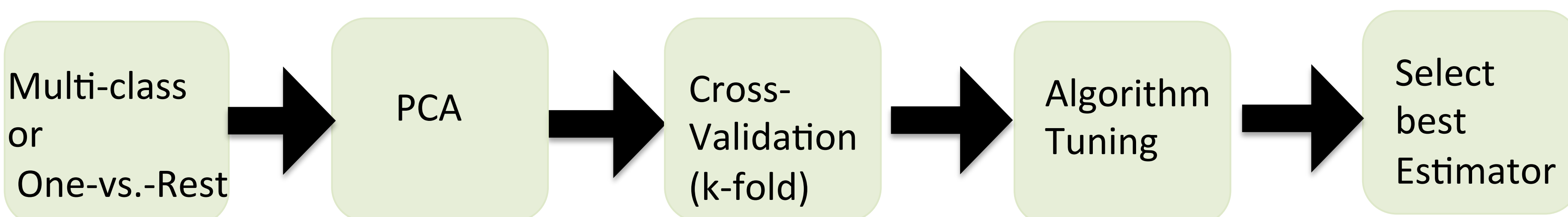
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2
0
2
4
Pca-two
Pca-one
Animal Class
Mammal
Bird
Reptile
Fish
Amphibian
Bug
Invetebrate

Approach

- Use all multi-class data or All V.S. One approach.
- Check if PCA improves algorithm performance.

or transformation, respectively.

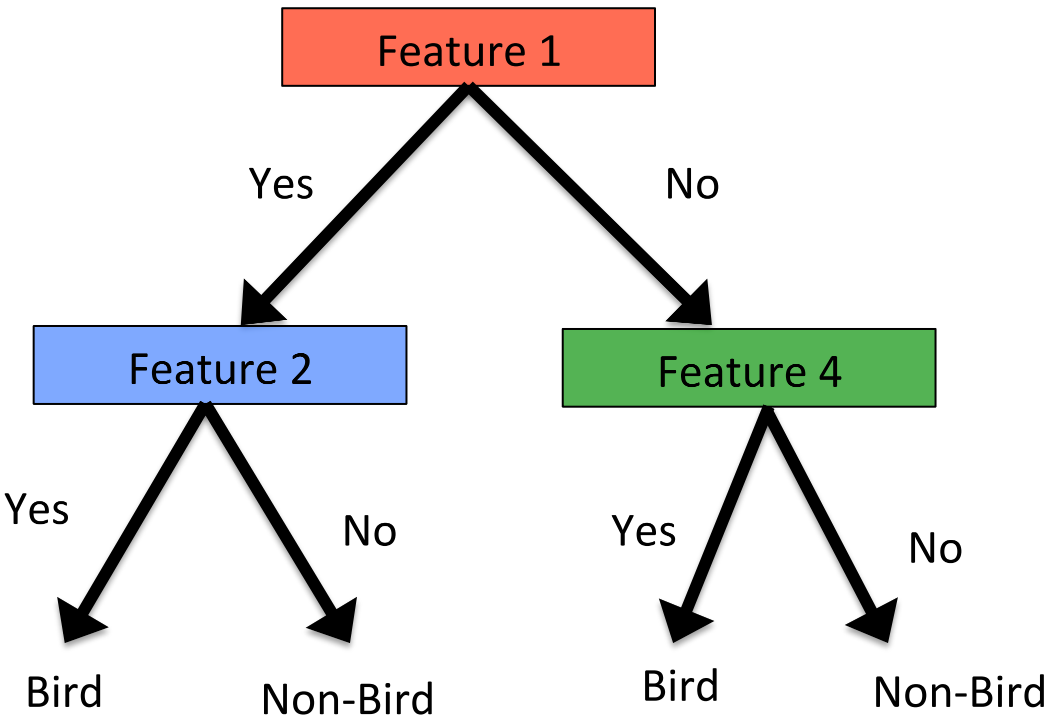
- Use (stratified) k-fold cross-validation to test different models
- Perform hyperdimensional tuning to find best estimators.



Model Selection

- Given the small size of the data set, complex ensemble model could be easily trained (Random Forrest).
- PCA Decomposition suggest that Cluster Models like K-Nearest-Neighbors could useful.
- An accuracy score (Jaccard similarity score) is utilized to measure performance of model.

Decision Tree



Cons:

- Weak Learner.
- Tend to overfit. (High Variance
- Sensitive to initial conditions.
- Non-Convex problem (ID3 aproach)

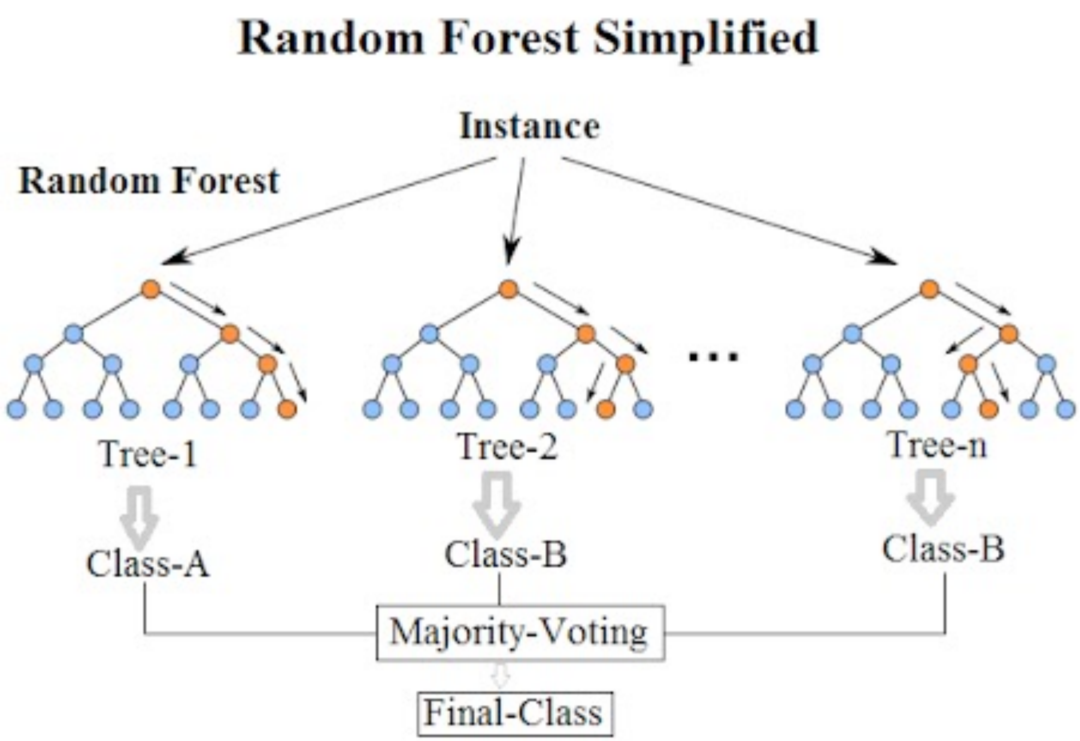
Pros:

- White Box.
- Easy to understand. (Visualization)
- No need of big samples for training.

*Gini Purity: $I_G(p) = \sum_{i=1}^{J} p_i(1-p_i)$, *J* are the different classes {Bird,Non-Bird}

Grus, Joel M. *Data Science from Scratch.* City:O'Reillly Media, Inc, 2015. Print.

Random Forest



◆ Ensemble approach

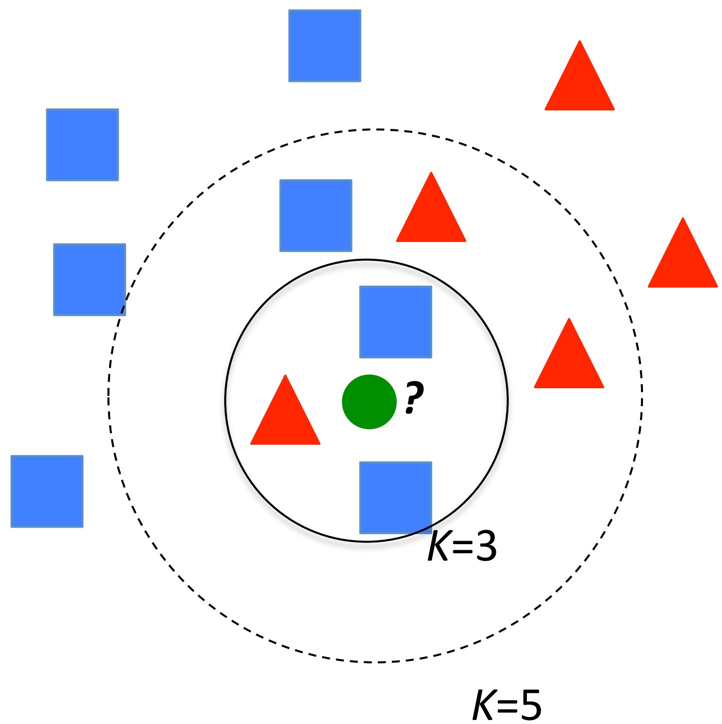
Where multiple Tree are Built

◆ Bagging Sampling.

◆ Feature Bagging.

https://www.kdnuggets.com/2017/10/random-forests-explained.html

K-Nearest Neighbors (KNN)



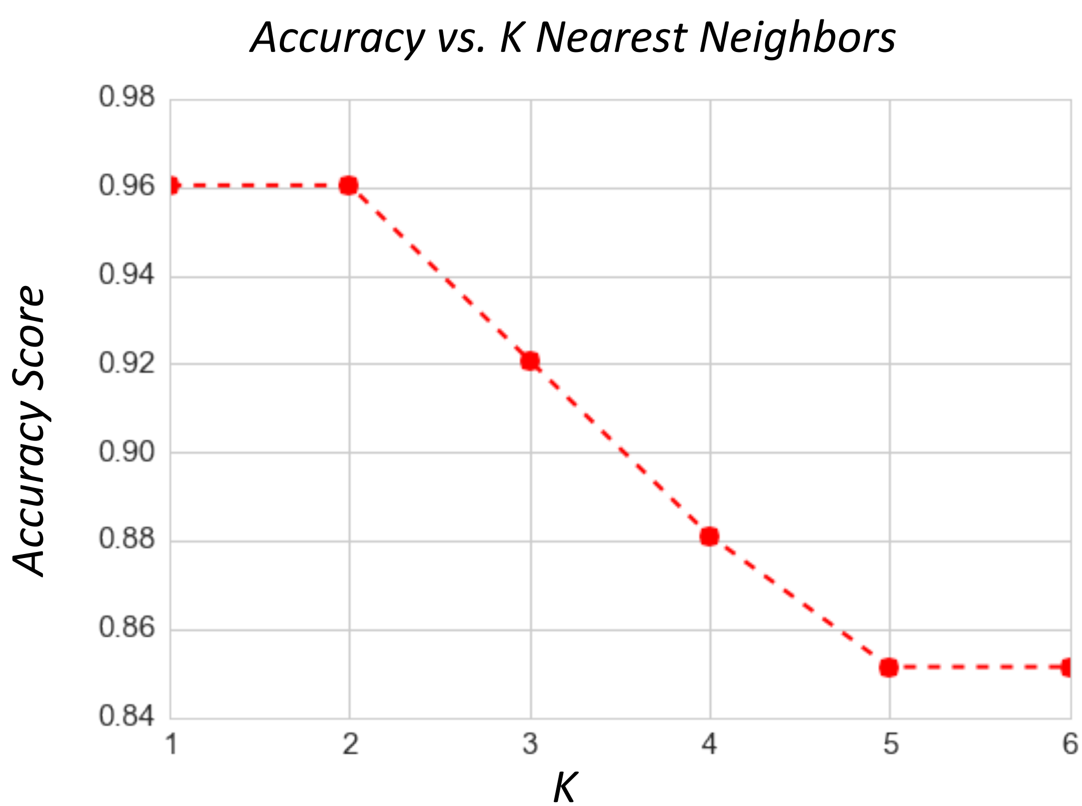
- Easy Model
- All that is needed is a metric and that points close to each other are similar

Sample Code (Pipeline)

```
In [97]: pipe = Pipeline([
             ('reduce_dim', SelectKBest()),
             ('classify', RandomForestClassifier())
         ])


         N_FEATURES_K = [10]
         N_ESTIMATORS = [2,3,4,5,6]
         MAX_FEATURES = [2,3,4,5]
         param_grid = [

             {
                 'reduce_dim': [SelectKBest(chi2)],
                 'reduce_dim__k': N_FEATURES_K,
                 'classify__n_estimators': N_ESTIMATORS,
                 'classify__max_features': MAX_FEATURES
             },
         ]

         grid_Chi2 = GridSearchCV(pipe, cv=3, n_jobs=1, param_grid=param_grid,return_train_score=True)

In [98]: grid_Chi2.fit(X_train, Y_train)

         mean_test_scores =np.array(grid_Chi2.cv_results_['mean_train_score'])
         print(mean_test_scores)
         print(np.mean(mean_test_scores))

[ 0.92011991  0.97512438  0.98037979  0.97489117  0.97986629  0.93997333
  0.96052637  0.96629528  0.97986629  0.97019633  0.94490137  0.96029316
  0.98037979  0.98507463  0.98507463  0.94982942  0.96160045  0.97517146
  0.97540467  0.98507463]
0.967502167858
```

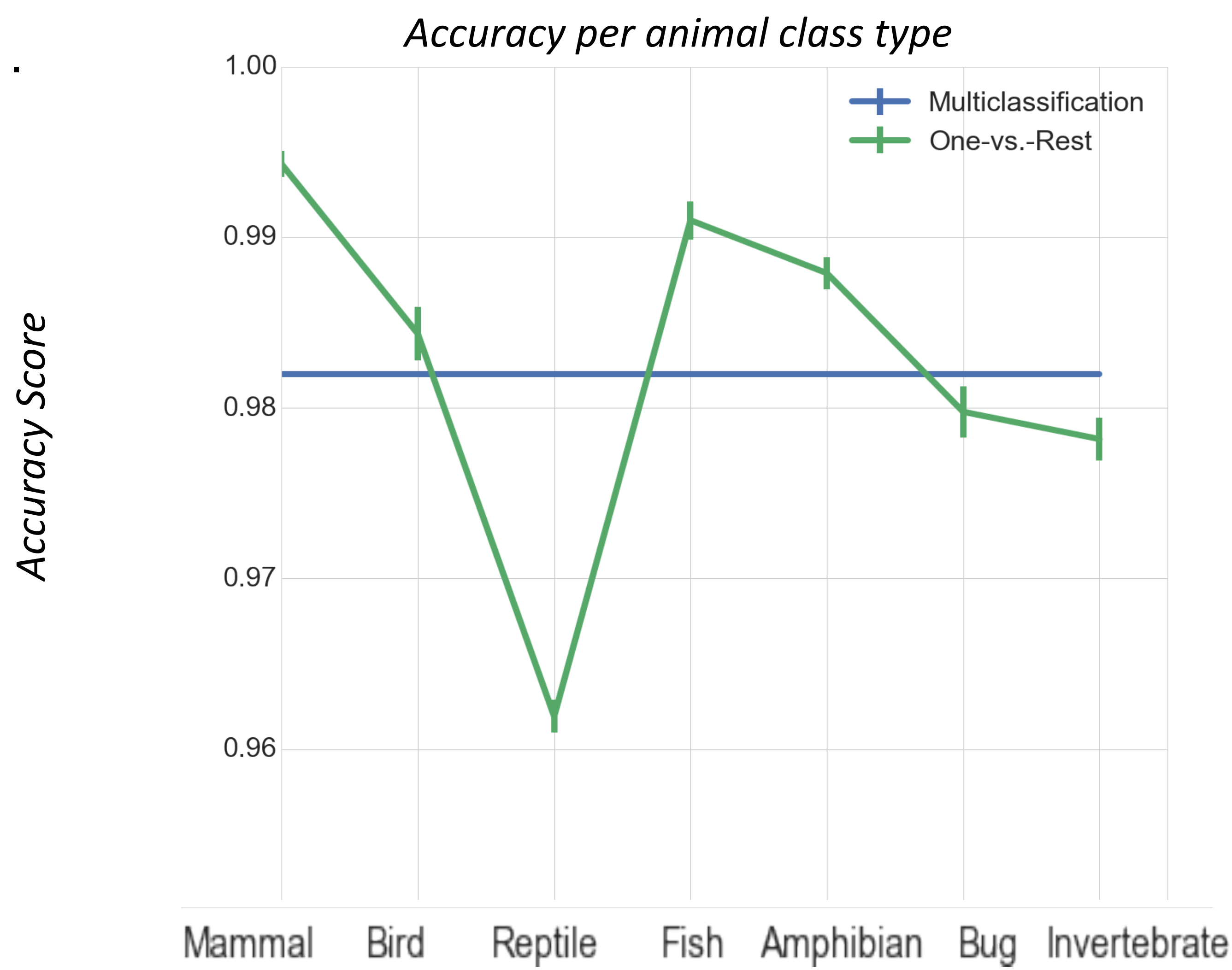
Multi-class Classification

- Random Forest performs better KNN similarly with scores =0.98 and 0.96, respectively.



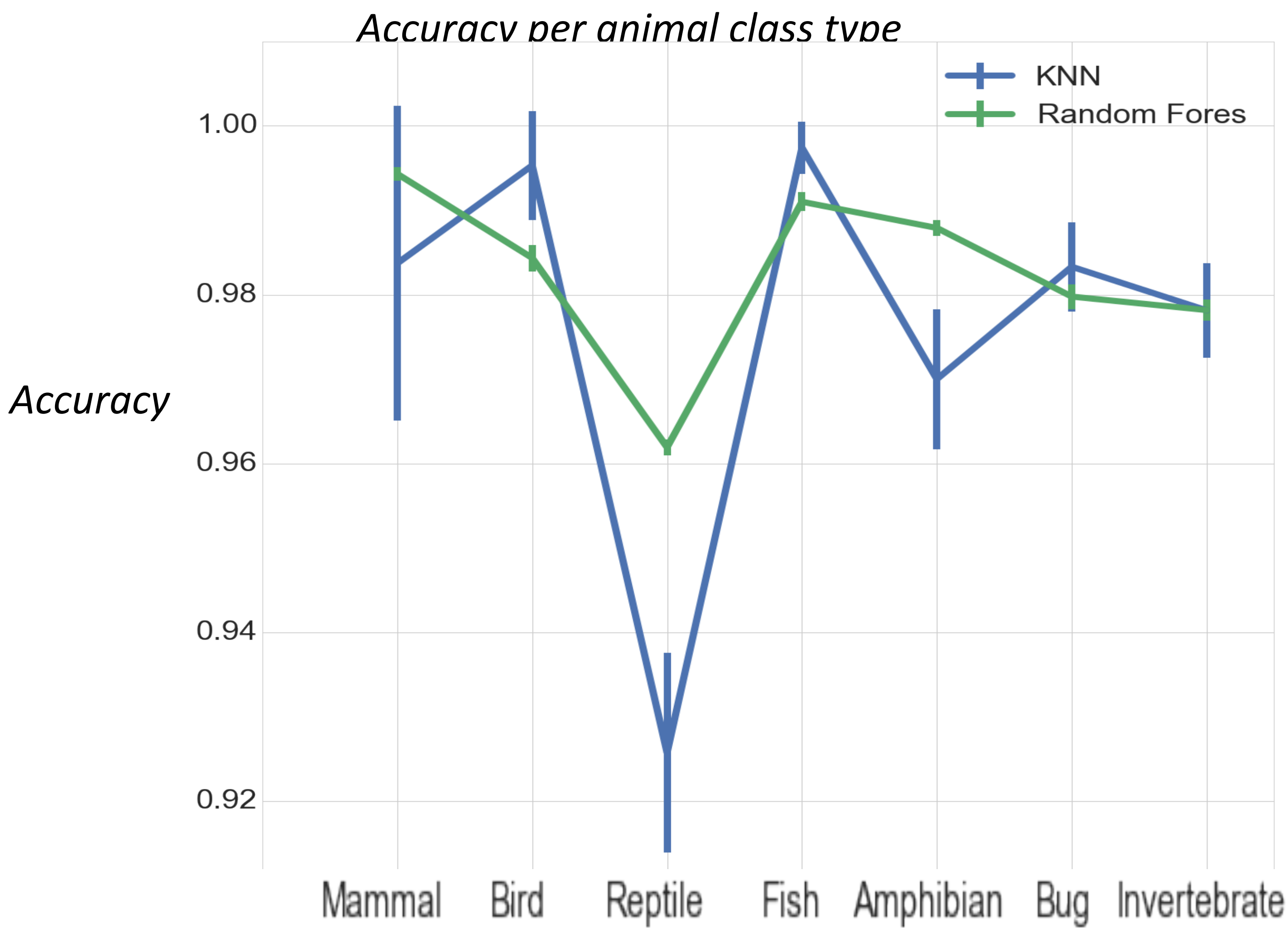
Multi-class Classification vs. One-vs. - Rest

◆ Random Forest showed same performance as Random Forest clustering .



Random Forest vs KNN
(One vs. Rest)

◆ One vs. Rest strategy showed better results than multiclass-classification.



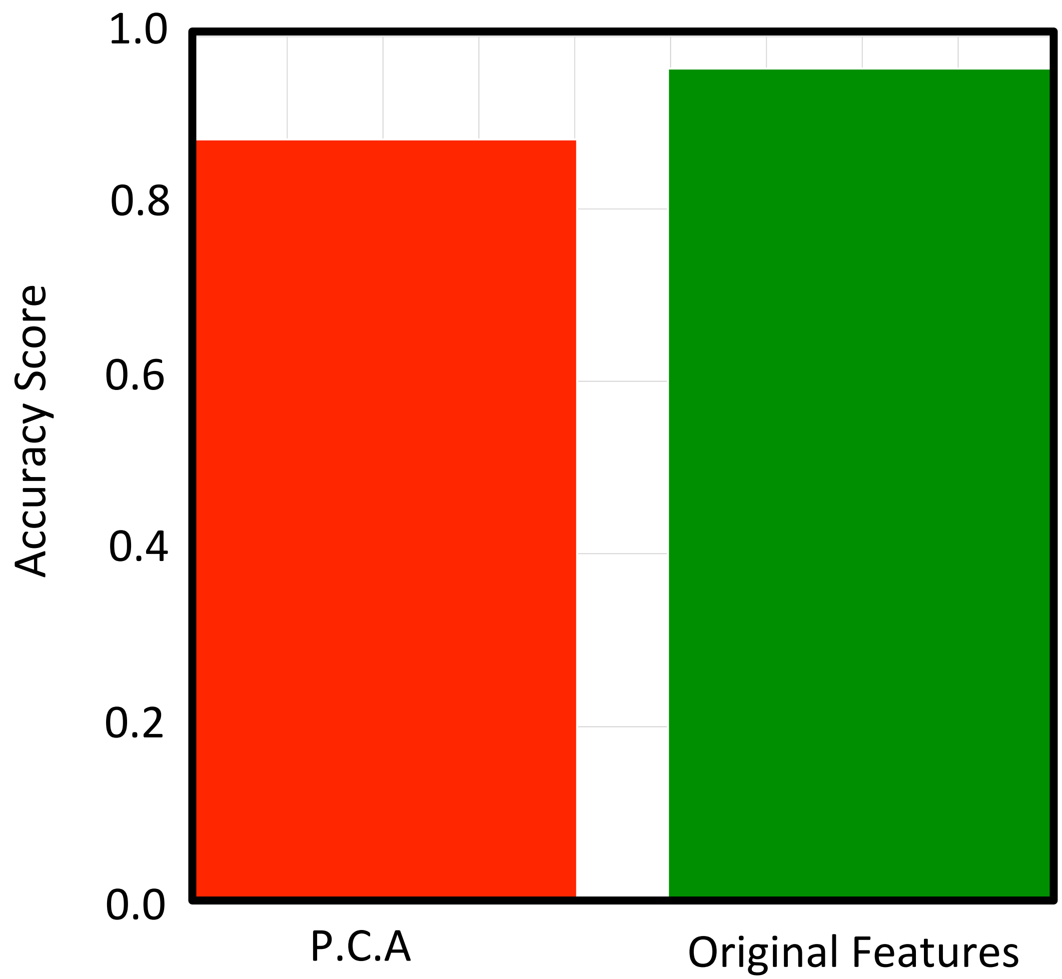
Feature Importance (Random Forest)

Amphibian	Hair	Feather	Eggs
Invertebrate	Backbone	Legs	Breathes
Fish	Fins	Breathes	Legs
Bird	Feathers	Legs	Tail
Mammal	Eggs	Hair	Milk
Reptile	Aquatic	Feathers	Toothed
Bug	Legs	Backbone	Tail

(*extracted form feature_importances attribute)

PCA vs. Original Features (Random Forest Multiclassification)

*PCA scores = 0.88 vs Chi-Square-test score = 0.97



Conclusion

- Original Features act as predictors.
- One vs.-rest strategy is better that multi-classification strategy.
- Feature importance depends of the class-type animal (One-vs.-rest strategy)