DD2324: Project 1

Eric Bojs

ABSTRACT

This report was written as the first assignment for the course DD2423 Machine Learning, Advance for the fall term 2020 at KTH Kungliga Tekniska Högskolan. The report cannot be independently, it is assumed that the reader has the project questions at hand.
Solutions to all seven problems are presented. The code can be found at the following github repository: HERE.

Keywords: None

PROBLEM 1

1. For every pair of eigenvector λ_i and eigenvalue u_i, we will consider the complex conjuagate of the equation system.

$$\begin{cases} Au_i = \lambda_i u_i \\ \bar{A}\bar{u}_i = \bar{\lambda}_i \bar{u}_i \end{cases}$$

We know since A is sem.pos.definite that $A \in \mathbf{R}^{n\times n}$ and $u_i \in \mathbf{R}^n$ which means $\bar{u}_i = u_i$, $\bar{A} = A$. Multiply each equation from the left by u_i^T

$$\begin{cases} u_i^T A u_i = u_i^T \lambda_i u_i \\ u_i^T A u_i = u_i^T \bar{\lambda}_i u_i \end{cases} \quad \Rightarrow \lambda u_i^T u_i = \bar{\lambda} u_i^T u_i \Leftrightarrow u_i^T u_i(\lambda - \bar{\lambda}) = 0$$

Meaning $\lambda = \bar{\lambda}$ which is only true if $Im(\lambda) = 0 \Rightarrow \lambda \in \mathbf{R}$

2. If $A = A^T$ with eigenvalues λ_i and eigenvector u_i. Then by looking at any two pairs i, j and we study $\lambda_i < u_i, u_j >$ and using definition of scalar product together with definition of eigenvalues/vectors:

$$\lambda_i < u_i, u_j > = < \lambda_i u_i, u_j > = < Au_i, u_j > = \{A = A^T\} = < u_i, Au_j > = < u_i, \lambda_j u_j > = \lambda_j < u_i, u_j >$$

Giving

$$0 = \lambda_i < u_i, u_j > - \lambda_j < u_i, u_j > = (\lambda_i - \lambda_j) < u_i, u_j >$$

Note that if two or more eigenvectors share an eigenvalue, we can always adjust those vectors by looking at the span of such a space and pick *new* orthogonal eigenvectors.

3. We prove this statement by assuming by that there is a negative eigenvalue $\lambda_i < 0$ with a corresponding eigenvector u_i for a positive semi-definite matrix A. I.e $|u_i| = 1$, $Au_i = \lambda_i u_i$. This leads to a contraction by:

$$0 \leq u_i^T A u_i = \{Au_i = \lambda u_i\} = u_i^T (\lambda u_i) = \lambda u_i^T u_i = \lambda ||u_i||_2^2 = \lambda$$
$$\lambda \geq 0$$

Thus proving the statement.

4.

$$D_{ij} = A_{ii} + A_{jj} - 2A_{ij} = ||\mathbf{v}_i - \mathbf{v}_j||_2^2 = \sum_{k=1}^{n} (v_{ik} - v_{jk})^2 = \sum_{k=1}^{n} v_{ik}^2 - 2v_{ik}v_{jk} + v_{jk}^2 = ||\mathbf{v}_i||_2^2 + ||\mathbf{v}_j||_2^2 - 2\sum_{k=1}^{n} v_{ik}v_{jk}$$

identify

$$\begin{cases} A_{ii} = ||\mathbf{v}_i||_2^2 \\ A_{jj} = ||\mathbf{v}_j||_2^2 \\ A_{ij} = \mathbf{v}_i^T \mathbf{v}_j \end{cases}$$

We have to convince ourselves that this is solveable for $\{v\}_{i=0}^{n}$. It is clear that this is possible since A is pos. semidefinite: $A_{ii} = \hat{e}_i A \hat{e}_i \geq 0$ and that v_i can simply be the i't column of D which is assumed to exist.

PROBLEM 2

Yes! Since performing SVD would give $X = U\Sigma V^T$. Then looking at PCA of XX^T and X^TX, we solve for

$$XX^T = U\Sigma V^T (U\Sigma V)^T = U\Sigma V^T V \Sigma^T U^T = \{\text{knowing X sym, and V*V=1}\} = U\Sigma^2 U^T$$

and similarly $X^TX = V\Sigma V^T$. i.e. performing SVD gives us U **and** V. How great!

PROBLEM 3

Looking at $\max_W Tr(Y^T WW^T Y)$ s.t W is orthogonal by construction of PCA. It's clear that we have to limit $||w_i||_2^2 = 1$, otherwise looking at $(kW) : k \in \mathbf{R}$ we get $Tr(Y^T(kW)(kW)^T Y) = k^2 Tr(Y^T WW^T Y)$, i.e. no clear max!

We use properties of $Tr(AB) = Tr(BA)$, decomposition $Y = UDV^T$ and the fact that $V^T = \mathbf{1}$ to get

$$Tr(Y^T WW^T Y) = Tr(V\Sigma U^T WW^T U\Sigma V^T) = Tr(V^T V\Sigma U^T WW^T U\Sigma) = Tr(\Sigma^2 U^T WW^T U) = Tr(\Sigma^2 (W^T U)^T W^T U) =$$

$$= Tr(\Sigma^2 (W^T U)^2) = \{\text{since } w_i \perp w_j (i.e. w_j^T w_i = 0) : \forall i \neq j\} = \sum_{i=1}^{k} \sigma_i^2 (w_i^T u_i)^2 = \sum_{i=1}^{k} \sigma_i^2 (||w_i||_2 ||u_i||_2 \cos\theta)^2$$

Since $||w_i||_i = ||u_i||_2 = 1$ we get the maximum when $\theta = \pi n : n \in \mathbf{Z}$ i.e. when $u_i || w_i$. Thus $w_i = u_i \Rightarrow W_k = U_k$. And plugging in our values and continuing on from before we get max:

$$\sum_{i=1}^{k} \sigma_i^2 (||w_i||_2 ||u_i||_2 \cos\theta)^2 = \sum_{i=1}^{k} \sigma_i^2$$

PROBLEM 4

We know that a squared distance matrix $D^{(2)}$ constructed from unknown data Y will have infinitely many possible solutions for Y. This, since if $\hat{Y}$ is a solution, the matrix $(R\hat{Y} + T)$, where R is a rotation and T is a translation, will also have the same squared distance matrix $D^{(2)}$, since it's invariant of distances! Thus, we can assume that we look for a solution $\hat{Y}$ where $y_1 = \bar{0}$, since any Y can be translated by $T = [-y_1, 0, \ldots, 0]^T$. Pretty nifty!

The squared distance from y_i to y_j is

$$d_{ij}^2 = ||y_i - y_j||_2^2 = y_i^T y_i + y_j^T y_j - 2y_j^T y_i = \{s_{ij} := y_i^T y_j\} = s_{ii} + s_{jj} - 2s_{ij}$$

Given that $s_{ij} = -\frac{1}{2}(d_{ij}^2 - s_{ii} - s_{jj})$ we want to prove this is equal to $s_{ij} = -\frac{1}{2}(d_{ij}^2 - d_{1i}^2 - d_{1j}^2)$.
It's easy to see that it's sufficient to show

$$s_{ii} + s_{jj} = d_{1i}^2 + d_{1j}^2$$

.

Thus,

$$d_{1i}^2 + d_{1j}^2 = s_{11} - 2s_{1i} + s_{ii} + s_{11} - 2s_{1j} + s_{jj} = 2(s_{11} - s_{1j} - s_{1i}) + s_{ii} + s_{jj}.$$

Following $s_{11} - s_{1j} - s_{1i} = 0$ for the claim to be true. Which we can convince ourselves of:

$$s_{11} - s_{1j} - s_{1i} = y_1^T y_1 - y_1^T y_j - y_1^T y_i = y_1^T (y_1 - y_i - y_j) = 0$$

Only because we know there exists a solution where $y_i = \bar{0}$ this since Y is invariant of translation, this will hold. Thus proving the claim!

PROBLEM 5

We know that the MDS embedding is found by through eigenvaluedecomposition of $S = U\Lambda U^T$. And $X_{MDS} = \mathbf{1}_k \Lambda^{1/2} U^T$

The PCA embedding is found by eigenvaluedecomposition of $Y = U\Sigma U^T$ and will be on the form $X_{PCA} = U_k Y$

It's clear that the embedding are the same iff $\Lambda^{1/2} = \Sigma$.

$$\begin{cases} Y^T Y = (\Lambda^{1/2} U^T)^T (\Lambda^{1/2} U^T) = U\Lambda U^T \\ Y^T Y = (U\Sigma U^T)^T (U\Sigma U^T) = U\Sigma^2 U^T \end{cases}$$

Thus since Λ a diagonal, $\Lambda^{1/2}$ exists and $\Sigma = \Lambda^{1/2}$!

In the case where Y is known, MDS would require us to compute S which would be an extra computational step compared to PCA.

PROBLEM 6

It's quite easy to imagine cases with irregular sampled data when looking at k nearest neighbors, one will end up with multiple disconnected networks. This will inevitably give a disconnect. In the one dimensional case with points $\mathbf{X} = [0, 1, 2, 8, 9, 10]^T$ with looking at 2-nearest neighbors, graphs of $\{1, 2, 3\}$ and $\{8, 9, 10\}$ appears. i.e. disconnected graphs.

One way to combat this is either by varying the parameter k until no disconnect appears. However, this can be quite inefficient as $k = N/2$ in the worst case. Another way to combat this is by systematically allowing some points to have more than k number of neighbors until all points are connected. This by finding two points in the disconnected networks which are the closest and connect those two.

PROBLEM 7

The project was carried out using a Jupyter Notebook due to its small size. The full *one page* Notebook can be found at [insert Github]

0) Data Preprocessing

The data was loaded into Python using the library Pandas. Index was set to `'animal_name'` as this should be unique. One double entry of `frog` was found and deleted from the set.

The parameter space was set to {`'hair', 'feathers', 'eggs', 'milk', 'airborne', 'aquatic', 'predator', 'toothed', 'backbone', 'breathes', 'venomous', 'fins', 'legs', 'tail', 'domestic', 'catsize'`]

The only parameter which was not boolean was `'legs'` which contained values of $\{0,2,4,5,6,8\}$. At first glance this could be seen as categorical. However, due to the nature of legs having a linear interpretation, i.e. animals having 2 legs are intuively *closer* to 4 than 6 and 8, the parameter was remapped to $\{0,1,2,0,3,4\}$. The $5 \to 0$ was done due to the only animal having five legs was 'starfish' is more seen as an abnormality and the definition of 'legs' is unclear. This assumption might not be perfect, but it is better than nothing.

We chose to standardize each column of the data by *both* subtract the mean and divide by the variance.

Notes on visualization: Since some of the animals had the exact same feature, any dimensionality reduction we perform the points will have the same mapping in the final form. Thus, when visualization the data, we only write out the name of the first of two or more animals which appears in the data set if the animals have the exact same parameter values.

i) PCA

The Principal Component Analysis was performed with the `scikit` library's function `decomposition.pca` to find $U_2 = [u_1, u_2]$

$$X_{PCA} = U_2 X$$

With u_1 and u_2 being the first and second principal component respectively. View Fig 1.

What is also worth looking is how much of the variance which is kept in the transformation. View Fig 2.

We see that at 2 components, we keep about 67% of the original variance within the data set.

ii) MDS

A distance matrix D from the data using ones favorite distance function $d : X \times X \to \mathbf{R}$. I looked at the Minkowski distances function for different values of p. SVD composition of the S which is found by double centering trick (see lecture notes) then yields our principal components and our final transformation.

In the classical MDS, $p = 2$, it will yield the *exact* same plot as the PCA done previously.

To infer the importance of different parameters k we simply look at $|u_{1k}$ and $|u_{2k}$. As the most influencial parameters will have a large absolute value.

Most influencial are:

	Parameter	Value
PC1	Milk	-0.442663
PC2	Fins	0.450397

We can look at different Minkowski distances, i.e. altering p when creating the distance matrix D^2. View Fig 3 some different MDS plots.

iii) Isomap

For the isomap implementation I first created a distance graph by only looking at a fixed k of nearest neighbors to each point. What should be noted is than due to the large amount of overlapping points, it's

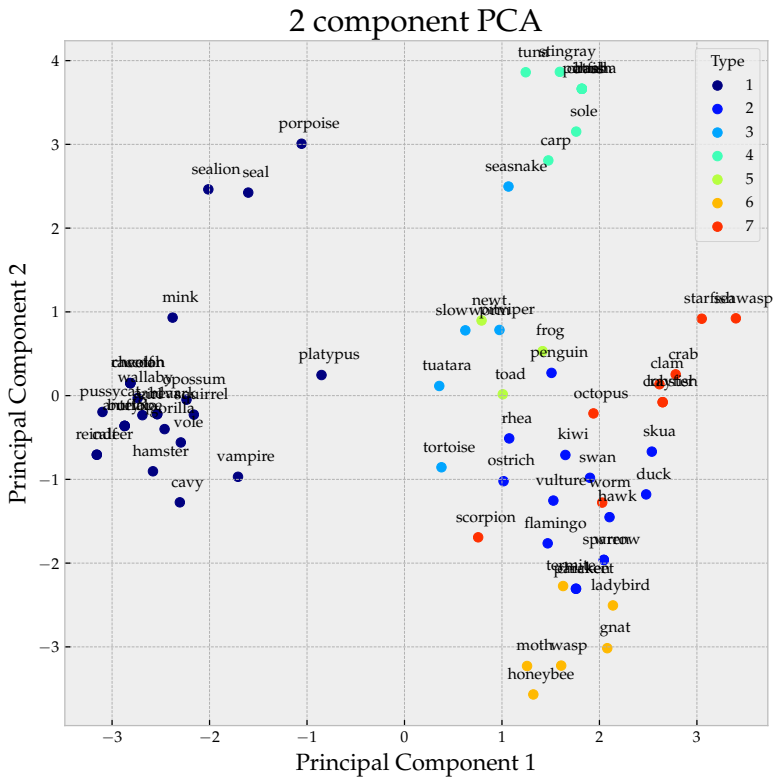

Figure 1. PCA

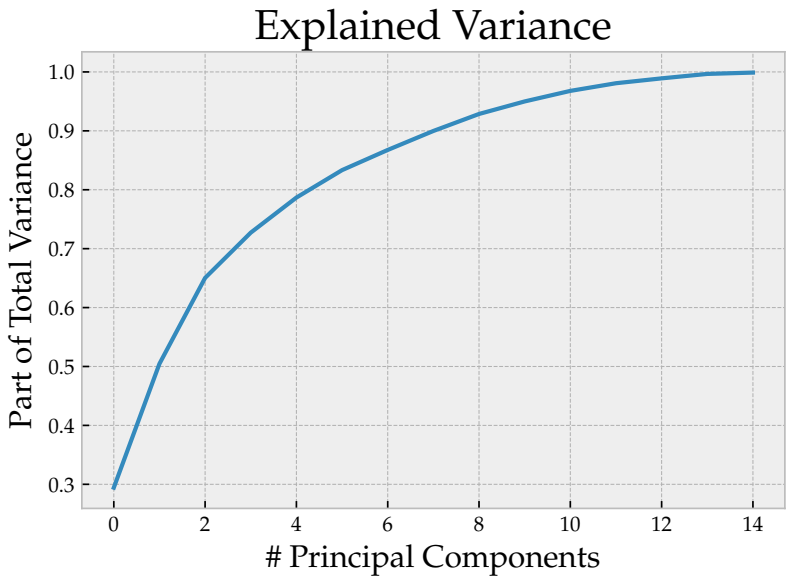

Figure 2. Percentage of Explained Variance kept

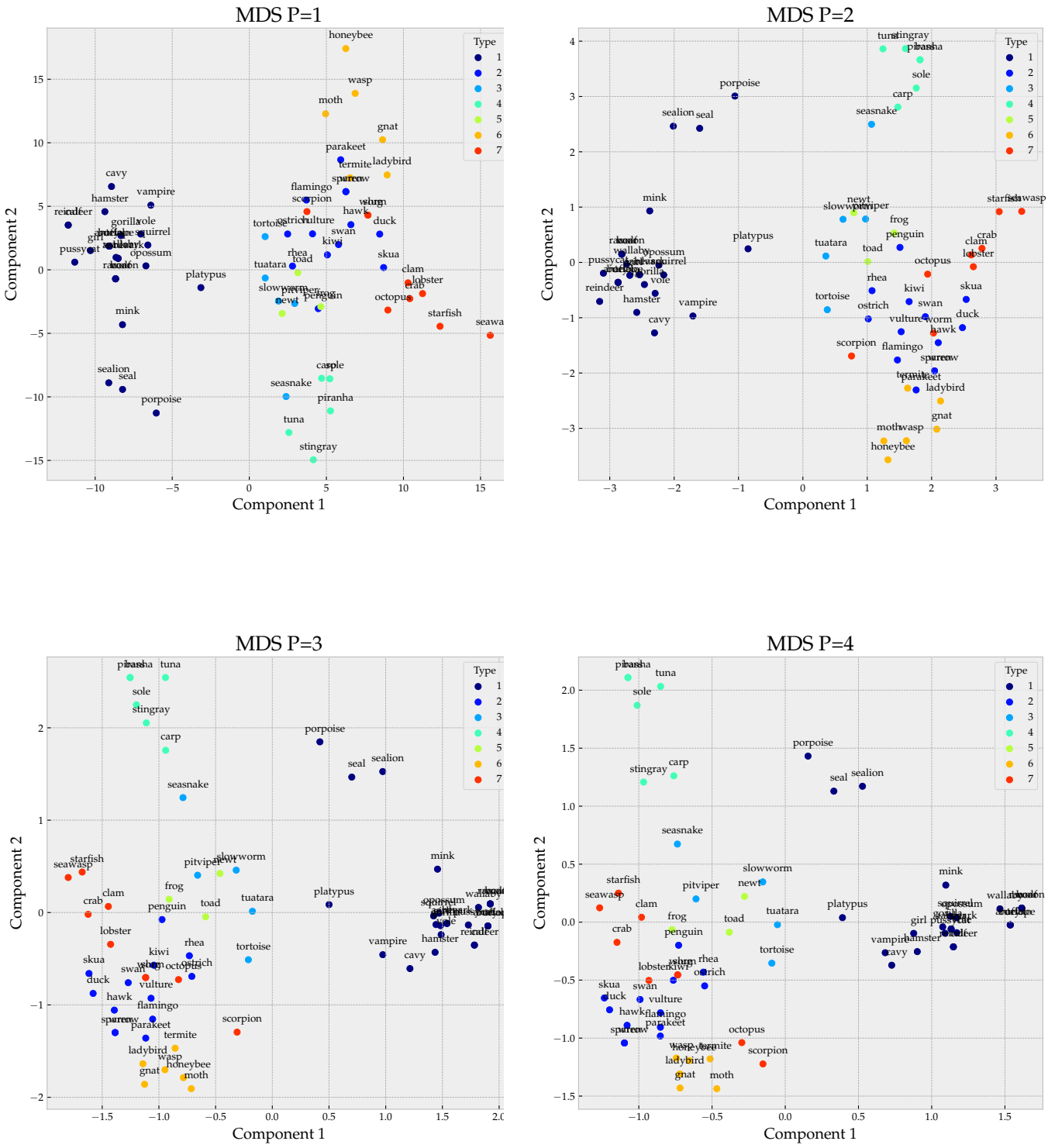

Figure 3. MDS for different values of p

better to look at the k nearest neighbors excluding the ones which are a distance of 0 away. If this is not done, we have a higher chance of getting disconnected graphs which is not sought for. From the distance graph we get the distance matrix from performing Floyd Warshall algoritm on the distance graph.

Getting our final S we simply used the double centering trick as described in the lecture notes.

One could technically optimize k for the smallest variation within each type, but we will simply look at some different values of k. Fig. 4 for plots.

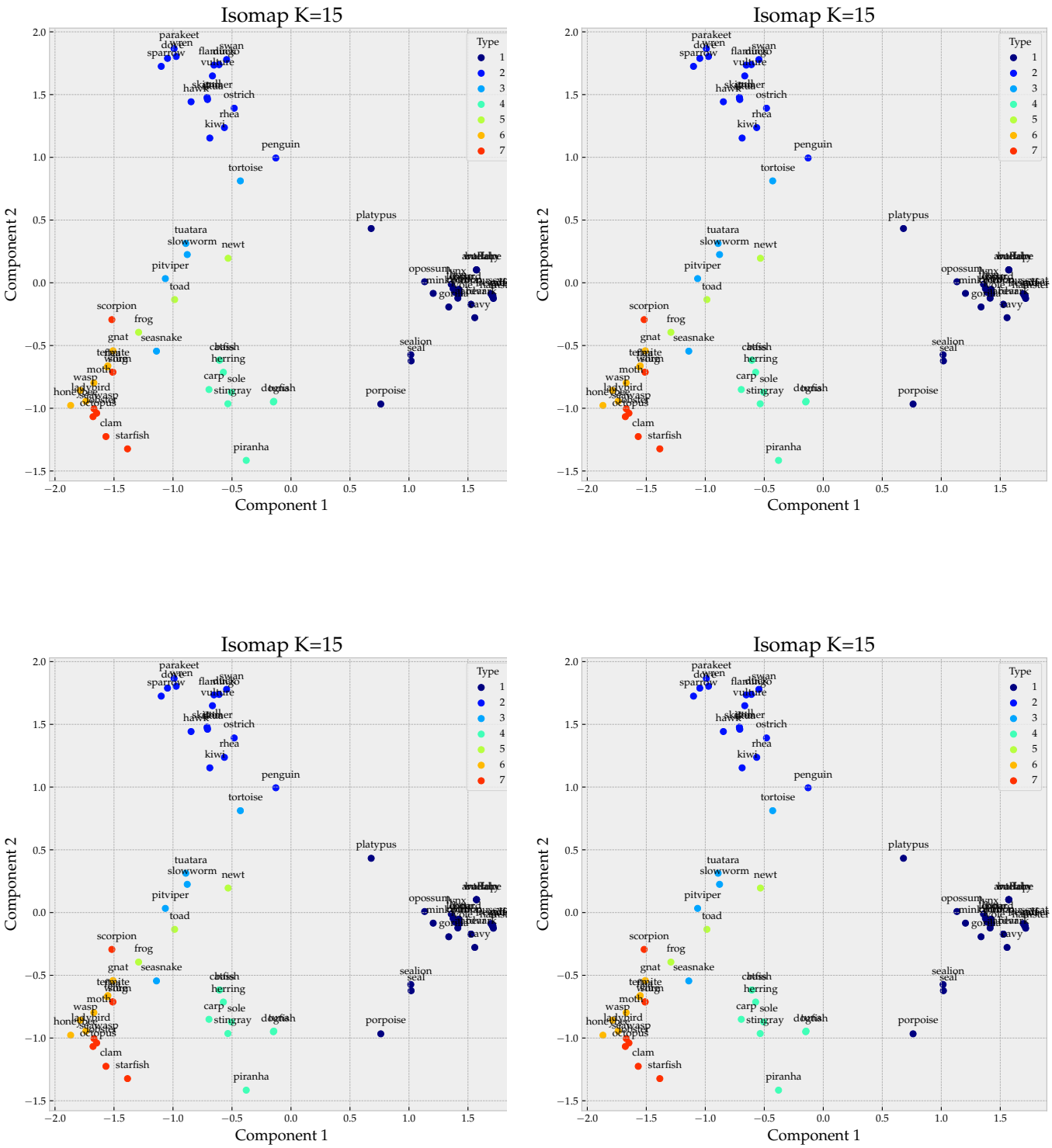

Figure 4. IsoMap for different values of k