Computational semantics - 2nd assignment:

Distributional representations for categorisation

Ana Mestre
Jonatan Koren

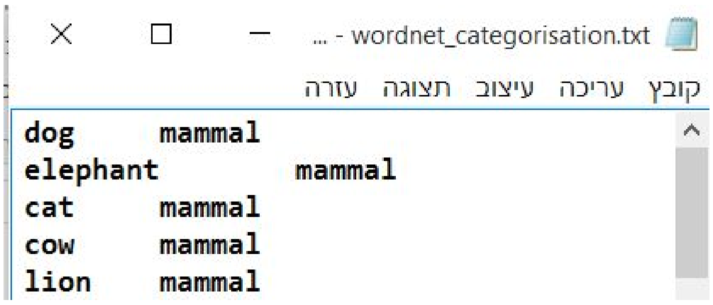
1) Categorisation of words

In this exercise, we used 2 files: categories_battig.txt and words_batting.txt, one used for given categories and the other for word pool.

First, we imported the files and then we did a word classification: we searched for synonyms for each word: the groupings of synonymous words that express the same concept. Some of the words have more than one 'synset'.

For each synonym, we adjusted its hypernym. In addition, for each such hypernym, we searched for its own lemmas group so that we could check whether one of the lemmas fits one of the given categories.

Finally, we created a file called "wordnet_categorisation.txt" and wrote the required words into it after verifying that no duplicate entries existed



```
dog     mammal
elephant        mammal
cat     mammal
cow     mammal
lion    mammal
```

For this exercise, we found that some words could have been linked into more than one category. However, for reasons of simplicity we decided to just use the first category we could match.

2) How the nouns in our mini-taxonomy get grouped in distributional space.

We now want to represent the words and categories in space so we need to use the file we created in section 1. We will present them in space using their vectors. In addition, their average vector called prototype vector.

First, we loaded the semantic model, then we extracted categories and words for each category, we separated the results from Exercise 1 into 2 dictionaries: one contains categories as keys and the other contains words as keys so we can effectively pull out what we need.

After defining which elements we want to display, we created T-SNE plots that use a t-distribution to measure points of similarity on the map after performing a dimensionality reduction from high to 2D (or 3D in other cases).

We created different colors for the elements - words, hypernyms, prototype categories. We refrained from displaying the labels for the words in order to clearly distinguish between the different categories.

We created 10 charts for 1000 iterations of T-SNE.

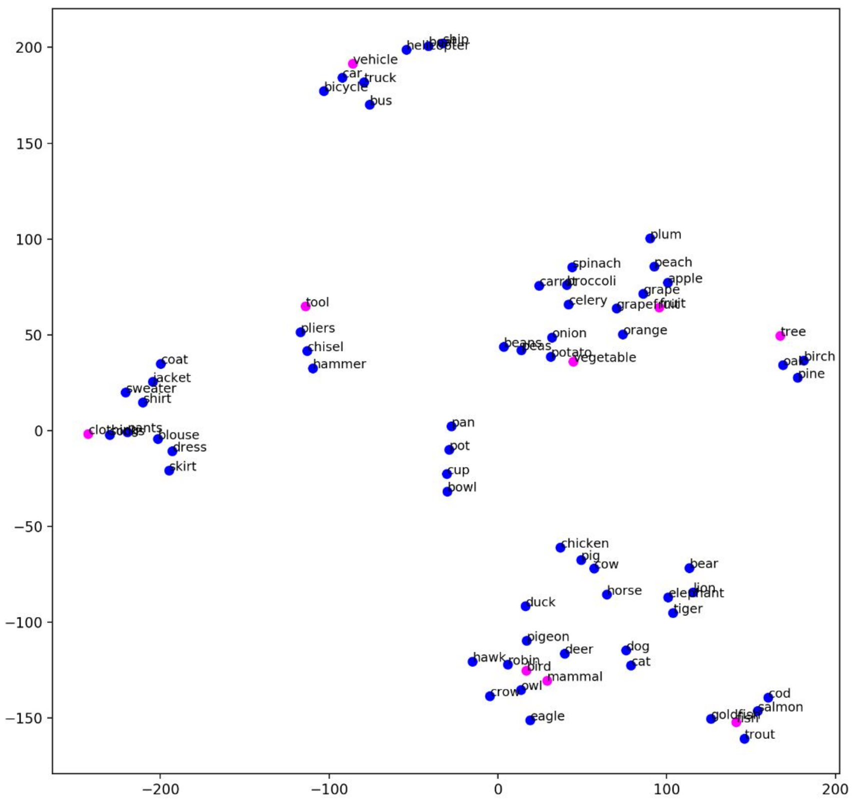
The iteration 100 has a large error and then decreasing error with jumps as a function of the number of iterations.

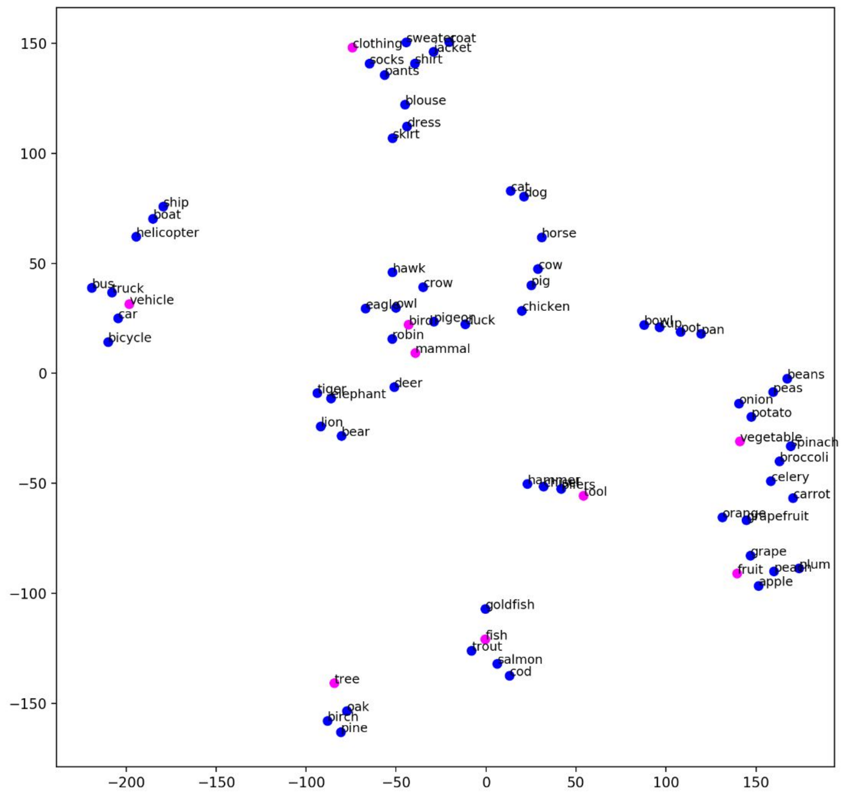
2A) Representation of the target words and their hypernyms

The following charts show the words divided by their respective categories using T-SNE, similarity of t distribution between vectors in space.

Note that in each run, categories move away and approach each other as well as words that move closer to words in another category. For example, at first the tree category is closer to fruits and vegetables, while at the second plot these two categories aren’t as close as before. Actually in the second graph, fish is closer to tree than it was on the first one. However, as a general tendency we can see that words that are related (that belong to the same category) are usually closer between them despite the graph.

There also has to be pointed out that some hypernyms representation such like kitchen_utensil and dishware are missing. This happened because the semantic model doesn’t include these words on it.


vehicle
car
truck
bicycle
bus
helicopter
boat
ship
plum
peach
apple
grape
grapefruit
fruit
orange
spinach
carrot
broccoli
celery
beans
peas
onion
potato
vegetable
tool
pliers
chisel
hammer
tree
oak
birch
pine
coat
jacket
sweater
shirt
clothing
socks
pants
blouse
dress
skirt
pan
pot
cup
bowl
chicken
pig
cow
horse
bear
lion
elephant
tiger
duck
pigeon
deer
dog
cat
hawk
robin
bird
mammal
crow
owl
eagle
cod
salmon
goldfish
fish
trout


clothing
sweater
coat
jacket
socks
shirt
pants
blouse
dress
skirt
ship
boat
helicopter
bus
truck
vehicle
car
bicycle
cat
dog
horse
cow
pig
chicken
hawk
crow
eagle
owl
bird
pigeon
duck
robin
mammal
deer
tiger
elephant
lion
bear
bowl
cup
pot
pan
beans
peas
onion
potato
vegetable
spinach
broccoli
celery
carrot
hammer
chisel
pliers
tool
orange
grapefruit
grape
fruit
peach
plum
apple
goldfish
fish
trout
salmon
cod
tree
oak
birch
pine

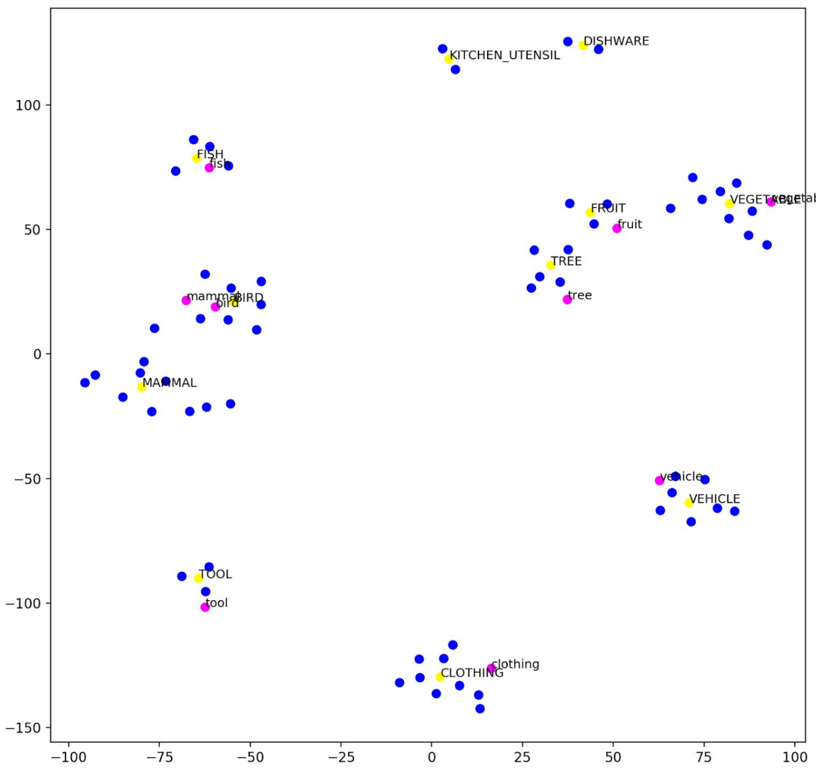
2B) Representation of words, their hypernyms and their prototype categories.
The following diagrams show prototype vectors that depict categories as averages of the target words.

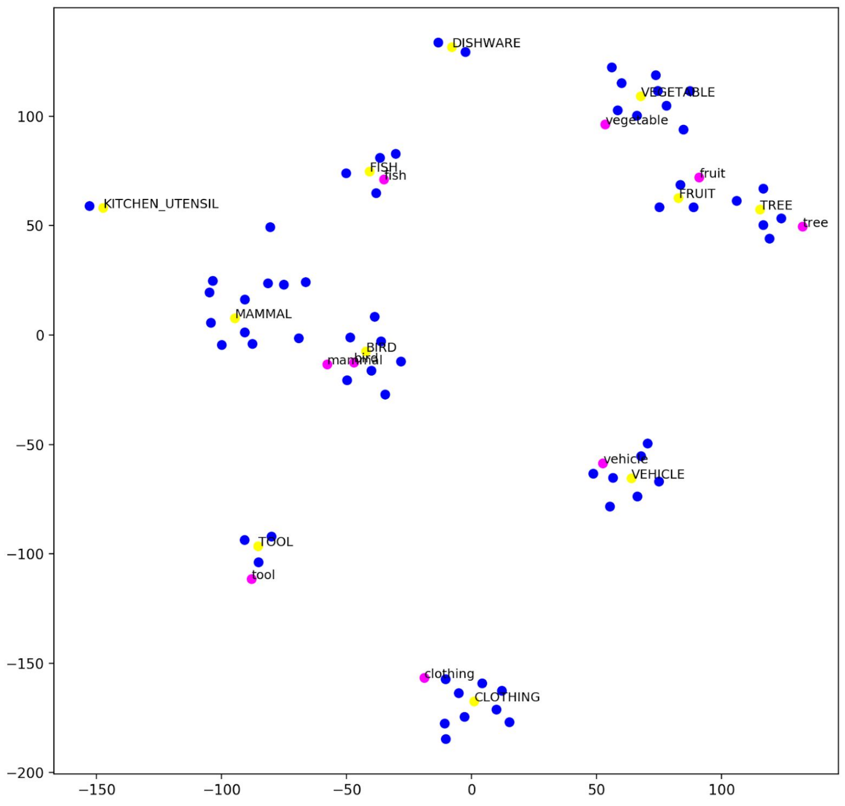
We can see how the locations for the prototype vectors change - approaching and moving away from each other. In some cases the category is close to the prototype vector (for example, bird and vegetable) but in some others is a bit far among themselves (like mammal). However, all pairs of hypernym-prototype category are still really close.

We can see in the different plots that the distance between every prototype category and its hypernym is always almost the same. This means that despite how our TSNE plot turns out, their closeness is clear. In addition, despite not having labels for all the words, we can see from the previous exercise and the dots distribution on this one, that the words belonging to a prototype category or the words who are hyponyms of a hypernym, stay close among themselves and close to the category and hypernym.

Processes of the type mentioned above are interesting and can indicate a strong connection between different types.

There also has to be pointed out that prototype categories such like kitchen_utensil and dishware don’t have their corresponding hypernym representation. This has happened because our model doesn’t have these words on it. However, since the prototype categories have been built by averaging its corresponding hyponyms, these can be represented.





In order to understand a little bit better the closeness of our taxonomy's words, the cosine similarity has been calculated for a hypernym and its prototype category and a hypernym and every single hyponym.

Find the results below:

```
--- Cosine similarities for hypernym: MAMMAL
------ with prototype category: MAMMAL , cos sim: 0.5736284
------ with hyponym: DOG , cos sim: 0.40285504
------ with hyponym: ELEPHANT , cos sim: 0.48828787
------ with hyponym: CAT , cos sim: 0.4389114
------ with hyponym: COW , cos sim: 0.35227427
------ with hyponym: LION , cos sim: 0.35374078
------ with hyponym: PIG , cos sim: 0.39698383
------ with hyponym: HORSE , cos sim: 0.25008544
------ with hyponym: BEAR , cos sim: 0.32408068
------ with hyponym: TIGER , cos sim: 0.49644738
------ with hyponym: DEER , cos sim: 0.4461344
--- Cosine similarities for hypernym: BIRD
------ with prototype category: BIRD , cos sim: 0.79547113
------ with hyponym: ROBIN , cos sim: 0.64826536
------ with hyponym: PIGEON , cos sim: 0.61340535
------ with hyponym: CHICKEN , cos sim: 0.42484805
------ with hyponym: EAGLE , cos sim: 0.47583324
------ with hyponym: DUCK , cos sim: 0.51255566
------ with hyponym: OWL , cos sim: 0.68258303
------ with hyponym: HAWK , cos sim: 0.4277546
------ with hyponym: CROW , cos sim: 0.5342829
--- Cosine similarities for hypernym: FISH
------ with prototype category: FISH , cos sim: 0.85716194
------ with hyponym: COD , cos sim: 0.6322998
------ with hyponym: GOLDFISH , cos sim: 0.53824615
------ with hyponym: SALMON , cos sim: 0.7214173
------ with hyponym: TROUT , cos sim: 0.77806544
--- Cosine similarities for hypernym: VEGETABLE
------ with prototype category: VEGETABLE , cos sim: 0.6754937
------ with hyponym: BROCCOLI , cos sim: 0.60931593
------ with hyponym: SPINACH , cos sim: 0.52789336
------ with hyponym: POTATO , cos sim: 0.6291501
------ with hyponym: ONION , cos sim: 0.5865841
------ with hyponym: CARROT , cos sim: 0.30947807
------ with hyponym: PEAS , cos sim: 0.47054243
------ with hyponym: CELERY , cos sim: 0.5443616
------ with hyponym: BEANS , cos sim: 0.44722337
--- Cosine similarities for hypernym: FRUIT
------ with prototype category: FRUIT , cos sim: 0.6719113
------ with hyponym: APPLE , cos sim: 0.64101475
------ with hyponym: ORANGE , cos sim: 0.3238018
------ with hyponym: GRAPE , cos sim: 0.6166846
```

```
--- Cosine similarities for hypernym: TREE
------ with prototype category: TREE , cos sim: 0.4965287
------ with hyponym: PEACH , cos sim: 0.35515893
------ with hyponym: PLUM , cos sim: 0.18013453
------ with hyponym: GRAPEFRUIT , cos sim: 0.25392604
------ with hyponym: OAK , cos sim: 0.46933174
------ with hyponym: PINE , cos sim: 0.42824233
------ with hyponym: BIRCH , cos sim: 0.43831939
--- Cosine similarities for hypernym: VEHICLE
------ with prototype category: VEHICLE , cos sim: 0.7262438
------ with hyponym: BOAT , cos sim: 0.42550415
------ with hyponym: CAR , cos sim: 0.78210956
------ with hyponym: SHIP , cos sim: 0.23633373
------ with hyponym: TRUCK , cos sim: 0.7027072
------ with hyponym: HELICOPTER , cos sim: 0.40108606
------ with hyponym: BUS , cos sim: 0.4596878
------ with hyponym: BICYCLE , cos sim: 0.48720223
--- Cosine similarities for hypernym: CLOTHING
------ with prototype category: CLOTHING , cos sim: 0.5712079
------ with hyponym: SHIRT , cos sim: 0.4196832
------ with hyponym: PANTS , cos sim: 0.44696242
------ with hyponym: SOCKS , cos sim: 0.5270721
------ with hyponym: JACKET , cos sim: 0.4443713
------ with hyponym: SWEATER , cos sim: 0.48365396
------ with hyponym: SKIRT , cos sim: 0.2864983
------ with hyponym: COAT , cos sim: 0.35933843
------ with hyponym: DRESS , cos sim: 0.48631454
------ with hyponym: BLOUSE , cos sim: 0.41050074
--- Cosine similarities for hypernym: TOOL
------ with prototype category: TOOL , cos sim: 0.2510983
------ with hyponym: HAMMER , cos sim: 0.116041705
------ with hyponym: CHISEL , cos sim: 0.21023247
------ with hyponym: PLIERS , cos sim: 0.2545492
```

For the cosine similarity results, we can see that the similarity between a hypernyms and its prototypes category gets higher as the similarity between this hypernyms and its hyponyms that is high as well. That was an expected behaviour since every prototype category vector is built as a result of averaging the vectors of its corresponding hyponyms.

In this case, we are getting high similarity values for fish and its hyponyms (and as explained before, with its prototype category too). However, for tools we are getting low values of similarity. From this low values we could think that maybe the hypernyms we had selected for some words weren’t the best option. Nevertheless, almost all similarities between hypernyms and prototype categories are > 50%. So our model and categorization has been moderately good.

2C) Clustering with K-means

We now clustered our target words with K-Means - given n observations {x1,x2,x3…,x11}, the observations are assigned to k clusters S={S1,S2,S3,…,S11} so as to minimize:

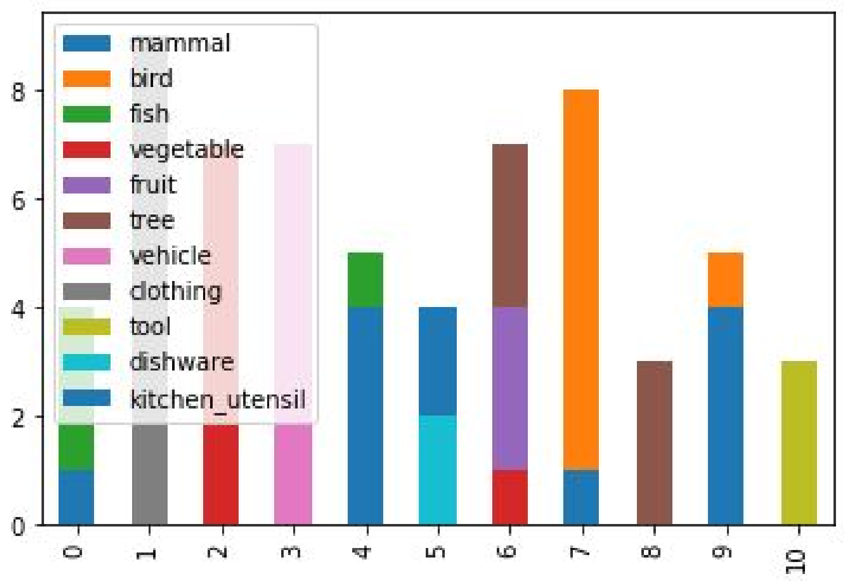
$$\underset{\mathbf{S}}{\operatorname{argmin}} \sum_{i=1}^{k} \sum_{\mathbf{x} \in S_i} ||\mathbf{x} - \mu_i||^2$$

* μ_i is the mean of the observations in S_i

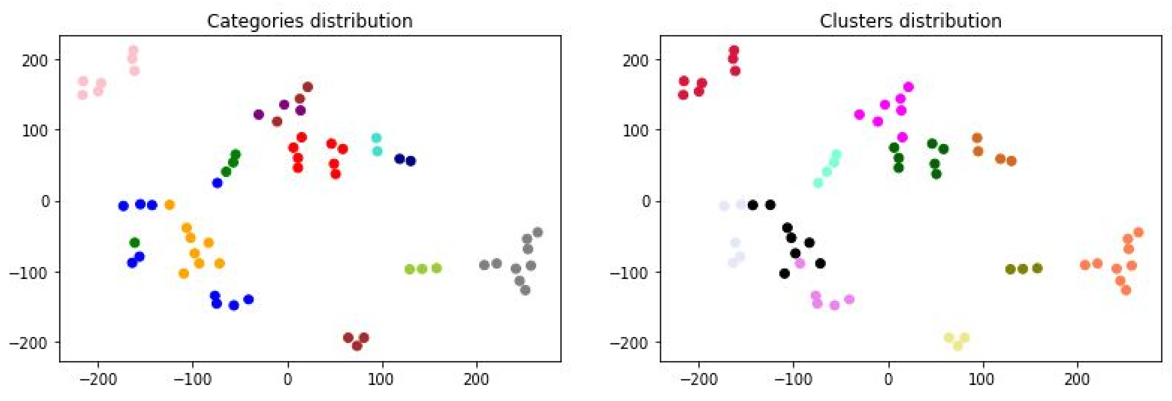
The table below represents every word linked to a cluster:

Clusters	Words
0	deer, cod, salmon, trout
1	shirt, pants, socks, jacket, sweater, skirt, coat, dress, blouse
2	broccoli, spinach, potato, onion, peas, celery, beans
3	boat, car, ship, truck, helicopter, bus, bicycle
4	dog, cat, cow, horse, goldfish
5	cup, bowl, pan, pot
6	carrot, apple, orange, grape, peach, plum, grapefruit
7	pig, robin, pigeon, chicken, eagle, duck, hawk, crow
8	oak, pine, birch
9	elephant, lion, bear, tiger, owl
10	hammer, chisel, pliers

In the following barplot we can see per each cluster (each bar) what categories fall into them. Like for instance, cluster 9 has 4 words from category mammal (from the table above, we can see that these are elephant, lion, bear, tiger) and 1 from the bird category (owl). While cluster 10 has three words corresponding to the tool category (hammer, chisel, pliers).



In the next plots, we can see first the real categorization of the words. This is the relation we made between words and categories on the first exercise. And in the second plot, we can see how our k-means algorithm separates the different dots.



In the first graph, the colours match the legend on the previous barplot. The second graph just has different colours to show how clustering separates the data. Nevertheless, in these plots, colours individually shouldn't be really taken into account. The important part is to see the changes after applying the clustering algorithm.

After taking a look at the previous results we can see that our clustering has some mistakes but hasn’t been that bad. In order to have a more precise measure to evaluate our results, we have used the purity operation, which tell us which clusters contain a single class. It counts the number of data points from the most common class in such cluster and then summarizing and averaging it’s total

$$\text{purity}(\Omega, \mathbb{C}) = \frac{1}{N} \sum_k \max_j |\omega_k \cap c_j|$$

For this we have obtained the contingency matrix of the k-means labels and the real labels (the real categories). This is our contingency matrix:

```
[[0 0 0 0 0 0 0 7 0 1 0]
 [0 9 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 2 0 0 0 0 0]
 [3 0 0 0 1 0 0 0 0 0 0]
 [0 0 0 0 0 0 3 0 0 0 0]
 [0 0 0 0 0 2 0 0 0 0 0]
 [1 0 0 0 4 0 0 1 0 4 0]
 [0 0 0 0 0 0 0 0 0 0 3]
 [0 0 0 0 0 0 3 0 3 0 0]
 [0 0 7 0 0 0 1 0 0 0 0]
 [0 0 0 7 0 0 0 0 0 0 0]]
```

Applying the purity formula, we obtained a purity value of 0.839. This value is quite high, which confirms that our prediction wasn’t bad, in fact it’s really good. Although some misjudgements have been done, the global tendency is pretty accurate.