16

K-means is ML unsupervised model to classify categorical data.

K is an integer number that stands for the number of centroids we wind up having. This number should denote the number of classes (or labels) that the data set is likely to belong to. The best ways to choose it are:

- prior to running the algorithm, knowing the number of classes the data should belong to (for instance, if we have a data set made of cats and dog, we want k = 2);
- empirically decide it using the elbow method.

Each input instance of the data set is tuple made of features which have to be numerical (in case these are categorical, like Strings, there way to translate them into numerical).

The algorithm is the following:

1. Assign the centroids at random;
2. For each element compute the L2 (or Euclidean distance) from each centroids;
3. Gather up in sets the element sharing a centroid as the closest one;
4. Recompute the centroids so that their location is at the (Euclidean) centre of all the nodes having it as the closest centroid.
5. If at least one centroid changed its location repeat from step 2

Now the model has been trained and these centroids indicate the centre of a class: whenever you want to classify a new input element you just have to compute the Euclidean distance from all the centroids, and the class of the closest one is the output.

17

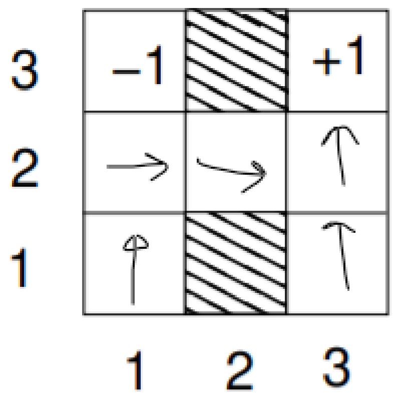
The issues I immediately come up with are about the bias. I would expect the model to be data-driven, as such trained by a CNN.
It is known that DL models use a black box, as such, we do not really know how they are making their decisions.
The result could be that there is an intrinsic bias in the input dataset, or that the DL algorithms introduced it, or that some supervising and feedback process did it.

As a result, there may bias against people wearing a hat, people with different skin tone, their gender etc.

So the 2 issues are that:

- the model may generate biases; and
- that you may not understand the model to fix these issues when they arise.

18



Briefly explain how policy iteration can be used to create a policy with
reference to computing π(1, 1).

The idea behind the policy iteration is to assign a score to each state based on:

- the possible moves it can operate;
- the probability of having the agent succeeding in the move it wants to operate (in other words, the determinism);
- the cost of each state (or the cost of making a move)

This process takes time as it you have to update the value of each cell in a sequence of iteration until convergence, but you end up having a nice score on each cell such that deciding the policy is pretty easy.

Say we have already operate the policy iteration, we should see a gradient ascend toward the positive reward +1.
In (1, 1) this flow should lead the agent to go either north or east in order to move closer to this reward, but as it cannot move east (since there is an obstacle), the policy $\pi(1, 1)$ will lead the agent to move north.

19

No idea what it is asking for, I will just write down the complete extensions.

CF: {}, a1, a2, a3, a4, a1 a4, a3 a4
A: a2, a4, a3 a4
C: {}. a4 a3, a2

20

	J	K	L
1	T	F	T
2	F	F	F
3	F	F	F

1 / 3

b. I do not know.