**Notes**

I have included my unstandardized code version for kNN implementation in the notebook in cell 2. It contains a lot of debug prints which can be used to see how it works behind the scenes. Although none of the data conflicted with the outcomes of the sklearn kNN function (uncomment print commands to compare outputs and set NNmin to 9 to make it easier to see) during implementation (on neighbours=10), for odd reasons my function gives out exactly same accuracy (I have not managed to find a seed where this is false) for each neighbour input (distance input differs), meaning the best result is always with NN=1. The reason for such behavior has eluded me to date.

**Results**

(seed=1234567)

|  |  |  |  |
| --- | --- | --- | --- |
| Clean data | Accuracy (%) | # neighbours | distance |
| Fold 1 | 96 | 1 | euclidean |
| Fold 2 | 93 | 1 | euclidean |
| Fold 3 | 93 | 1 | euclidean |
| Fold 4 | 100 | 1 | euclidean |
| Fold 5 | 96 | 1 | euclidean |
| total | 96±2 |  |  |
| Noisy data | Accuracy (%) | # neighbours | distance |
| Fold 1 | 90 | 1 | manhattan |
| Fold 2 | 86 | 1 | euclidean |
| Fold 3 | 76 | 1 | euclidean |
| Fold 4 | 73 | 1 | euclidean |
| Fold 5 | 90 | 1 | euclidean |
| total | 83±6 |  |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Clean | | | Noisy | | |
| 50 | 0 | 0 | 50 | **0** | 0 |
| 0 | 47 | 3 | 1 | 36 | 13 |
| 0 | 3 | 47 | 0 | 11 | 39 |

**Confusion matrices**

**Result Analysis**

1. It seems that my algorithm always gives out the same accuracy for whichever number of neighbours is chosen.
2. The choice almost never changes except for fold 1 in noisy data where manhattan distance gives better results.

**Questions**

1. When plotting without noise, the data is neatly distributed with clearly visible and distinct (most of the times) class areas. After adding noise a lot of dots are becoming mixed and often it is difficult to distinguish which area they belong to.