Exam Defi IA

Ziqi XU

Theoretical

Question 1

Dataset 1: DBSCAN should be used because the density of each dataset is quite different and there are some noise points.
Dataset 2: K-means algorithm will be efficient because the datasets are clearly separable, so that we can define the number of classes at the beginning
Dataset 3: Hierarchical clustering will be efficient because the datasets are separable, but it is difficult to tell the number of classes.

Question 2

Positive and negative entities should be equal (or nearly) in your dataset, otherwise the algorithm does not have much information on the positive class. In addition, when separate training and test dataset, make sure that there are around 50:50 positive and negative entities. Since you have 71 features, consider applying PCA or LDA (linear discriminant analysis) to reduce the number of features. Accuracy is one of the most useful measures, you may also look at the F1 score.

Question 3

Method 1: SVM with Gaussian Kernel. For this non linearly separable problem, using Gaussian Kernel can project the 2D data into 3D (or higher dimension), which makes it possible to find a hyperplane to separate the data.
Method 2: Random Forest. This algorithm is suitable for non linear problems.

Question 4

Continuous means there are a lot of (infinitive) possible states, and episodic means that the states are already known (for example, a maze).

Classification of religions

Pre-processing

In the dataset, most of the data are numerical, except for mainhue, topleft and botright. I have implemented two ways to transform them into numerical.
Method 1: using LabelEncoder in sklearn to automatically transform each column into numerical, then replace the original column.
Method 2: using get_dummies in pandas to transform each category into several, for example, mainhue into mainhue_red, mainhue_green, etc.

Classification

I have used SVM and Random Forest to classify the data. Here are the results.
We can see that Random Forest is better than SVM (Figure 3 vs Figure 1/2), and pre-processing method 1 is better than method 2 (Figure 3 vs Figure 4). However, the results for category 3, 4, 7 are not satisfying. I have checked the value counts for religions, and found out that there are too few examples in these categories (Table 1), which means that the algorithm does not have enough information during the training.

1	60
0	40
2	36
5	27
6	15
3	8
7	4
4	4

Table 1 Value counts for religion

I have searched for some methods to tackle with this problem, by using oversampling we can increase the number of data in category 3, 4, 7. There is a package imbalanced-learn that can help us. By using RandomOverSampler, which generate new samples by randomly sampling with replacement the current available samples (so that in each category there are 60 samples), I have obtained the results in Figure 5 (with Random Forest). The result is better than the previous ones.
This package also offers some advanced algorithms like SMOTE and ADASYN, but I did not manage to implement them as there are some errors in the format of our data.

Conclusion

Oversampling seems a good way to solve the problem of imbalanced dataset. Random Forest is by far the best solution to our problem. To achieve better result, we should also find a way to reduce the number of features.

	precision	recall	f1-score	support
0	0.58	0.64	0.61	11
1	0.49	0.85	0.62	20
2	0.75	0.40	0.52	15
3	0.00	0.00	0.00	2
4	0.00	0.00	0.00	1
5	0.00	0.00	0.00	8
6	0.00	0.00	0.00	1
7	0.00	0.00	0.00	1
micro avg	0.51	0.51	0.51	59
macro avg	0.23	0.24	0.22	59
weighted avg	0.46	0.51	0.46	59

Figure 1 SVC with kernel rbf

	precision	recall	f1-score	support
0	0.42	0.45	0.43	11
1	0.48	0.55	0.51	20
2	0.75	0.40	0.52	15
3	0.00	0.00	0.00	2
4	0.00	0.00	0.00	1
5	0.25	0.12	0.17	8
6	0.17	1.00	0.29	1
7	0.00	0.00	0.00	1
accuracy			0.41	59
macro avg	0.26	0.32	0.24	59
weighted avg	0.47	0.41	0.41	59

Figure 2 SVC with kernel linear

	precision	recall	f1-score	support
0	0.62	0.45	0.53	11
1	0.65	0.75	0.70	20
2	0.83	0.67	0.74	15
3	0.00	0.00	0.00	2
4	0.00	0.00	0.00	1
5	0.50	0.38	0.43	8
6	0.10	1.00	0.18	1
7	0.00	0.00	0.00	1
micro avg	0.58	0.58	0.58	59
macro avg	0.34	0.41	0.32	59
weighted avg	0.62	0.58	0.58	59

Figure 3 Random Forest, max feature 4

	precision	recall	f1-score	support
0	0.67	0.55	0.60	11
1	0.59	0.80	0.68	20
2	0.69	0.60	0.64	15
3	0.00	0.00	0.00	2
4	0.00	0.00	0.00	1
5	0.33	0.12	0.18	8
6	0.14	1.00	0.25	1
7	0.00	0.00	0.00	1
accuracy			0.56	59
macro avg	0.30	0.38	0.29	59
weighted avg	0.55	0.56	0.53	59

Figure 4 Random Forest, max feature 4, Method 2

	precision	recall	f1-score	support
0	0.58	0.69	0.63	16
1	0.55	0.33	0.41	18
2	0.92	0.86	0.89	14
3	0.90	1.00	0.95	18
4	1.00	1.00	1.00	22
5	0.76	0.72	0.74	18
6	0.82	1.00	0.90	18
7	1.00	1.00	1.00	20
accuracy			0.83	144
macro avg	0.82	0.83	0.82	144
weighted avg	0.82	0.83	0.82	144

Figure 5 Random Forest, max feature 4, oversampling