Data analysis for heart disease prediction present with the Cleveland dataset

Student: Yen-Hsing Li

Date: 2017/1/17

Outline

- Analysis software
- The dataset
- Processed with missing value
- Entropy
- Predict
- Conclusion

Analysis software

- Using Python for data analysis

- *Sklearn support data splitting used for predicting*
- *More flexible compared to R*
- *Currently used for research*

- In this program, it has several functions and goal
- *Missing value processing*
- *Entropy counting :evaluate entropy for each dimension and the whole dataset*
- *the confusion matrix*

- **Goal: trying to get more relative and lesser columns for predicting heart disease.**

The dataset

- This database contains 76 attributes, but all published experiments refer to using a subset of 14 of them.

- Expected to find the most related dimension subsets for predicting heart disease.

Attribute Information:

Only 14 attributes used:
1. #3 (age)
2. #4 (sex)
3. #9 (cp)
4. #10 (trestbps)
5. #12 (chol)
6. #16 (fbs)
7. #19 (restecg)
8. #32 (thalach)
9. #38 (exang)
10. #40 (oldpeak)
11. #41 (slope)
12. #44 (ca)
13. #51 (thal)
14. #58 (num) (the predicted attribute)

The dataset

age	sex	cp	trestbps	chol	fbs	restecg
63	1	1	145	233	1	2
67	1	4	160	286	0	2
67	1	4	120	229	0	2

thalach	exang	oldpeak	slope	ca	thal	num
150	0	2.3	3	0	6	0
108	1	1.5	2	3	3	2
129	1	2.6	2	2	7	1

Processed with missing value

- In this dataset, it has roughly 6 missing values in the 12 and the 13 column.
- Replaced with the means along the axis.

```
def processlosevalue(data):

    for i in range(len(data)):
        for j in range(len(data[i])):
            if(data[i][j]=="?"):
                data[i][j] = np.nan
    np.asarray(data)

    imp = Imputer(missing_values='NaN', strategy='mean', axis=0)
    imp.fit(data)
    y = imp.transform(data)

    return np.array(y,"int")
```

```
['38', '1', '3', '138', '175', '0', '0', '173', '0', '0', '1', '?', '3', '0']
[ 38   1   3 138 175   0   0 173   0   0   1   0   3]
```

Entropy

- We use entropy to represent the average information obtained from a single sample X .

$$H(X)=\sum_{i=1}^{q} p_i I(p_i)=\sum_{i=1}^{q} p_i \log_2 \frac{1}{p_i}=-\sum_{i=1}^{q} p_i \log_2 p_i$$

- Low entropy means X is from varied distribution, and so the values sampled from it would be more predictable.

- For Cleveland dataset , there are 13 columns ,the last column is target (0 to 4). We counted each target value’s entropy, and trying to get the uncertainty of the information.

Entropy

- From the figure shown above, we obtain that the[2,6,7,9,11,13]columns have low entropy(lower than 1)

```python
def entropy(mdict,count):
    m_entropy = 0
    for r in mdict:
        pk = mdict[r] / count
        m_entropy += -(pk * np.log(pk))
    print('m_entropy',m_entropy)
```

```
dimemsion 0 entropy 3.49734794908
dimemsion 1 entropy 0.685692588448
dimemsion 2 entropy 1.28021214464
dimemsion 3 entropy 3.11511024824
dimemsion 4 entropy 4.51644317792
dimemsion 5 entropy 0.405405991789
dimemsion 6 entropy 0.712397960422
dimemsion 7 entropy 3.99966452563
dimemsion 8 entropy 0.405405991789
dimemsion 9 entropy 0.771456497085
dimemsion 10 entropy 0.802313014536
dimemsion 11 entropy 0.640911469562
dimemsion 12 entropy 0.642743968904
candidate [1, 5, 6, 8, 9, 10, 11, 12]
```

```
dimemsion 0 entropy 3.03519729678
dimemsion 1 entropy 0.500402423538
dimemsion 2 entropy 0.567260893626
dimemsion 3 entropy 2.91637206583
dimemsion 4 entropy 3.51573965117
dimemsion 5 entropy 0.537544412474
dimemsion 6 entropy 0.760439096594
dimemsion 7 entropy 3.3573060099
dimemsion 8 entropy 0.642912439666
dimemsion 9 entropy 1.6623041931
dimemsion 10 entropy 0.839032681321
dimemsion 11 entropy 1.31920358024
dimemsion 12 entropy 0.582425687196
candidate [1, 2, 5, 6, 8, 10, 12]
```

Predict

- Using Naive Bayes classifier and decision tree
- Taking the [2,6,7,9,11,13]columns for analyzing

Naive Bayes classifier

```
diagonal sum:56
confusion_matrix:
[[35  0  1  0 15]
 [ 5  0  0  0 12]
 [ 1  0  2  1  8]
 [ 1  0  1  1 12]
 [ 0  0  0  1  3]]
```

decision tree

```
diagonal sum:49
confusion_matrix:
[[28 14  6  4  0]
 [ 3  5  5  3  0]
 [ 4  3  0  1  0]
 [ 3  3  5  5  0]
 [ 1  0  4  2  0]]
```

- The 1~4 classify attribute is easy to be misclassified
- Replace with all 1(> 50% diameter narrowing)

Predict

Naive Bayes classifier

```
[[76 75 74 75 78]
 [78 79 77 72 66]
 [67 73 77 71 72]
 [70 66 71 72 74]
 [67 71 70 72 74]
 [69 70 75 68 71]
 [71 72 66 73 73]
 [73 71 78 74 71]
 [73 70 69 76 68]
 [73 70 72 74 77]]
d_list length =50
max relative index = 6 diagonal_sum = 79
72.28
```

```
[[48 12]
 [13 26]]

diagonal sum:74
```

decision tree

```
[[83 81 82 82 79]
 [75 83 79 80 78]
 [77 86 76 84 77]
 [82 80 81 83 75]
 [79 86 82 79 79]
 [81 88 86 79 83]
 [80 76 77 82 79]
 [81 90 83 82 78]
 [72 81 84 77 75]
 [83 80 80 72 86]]
d_list length =50
max relative index = 36 diagonal_sum = 90
80.46
```

```
[[48  6]
 [13 32]]

diagonal sum:80
```

Predict

- Compared to original analyzing[1~13columns] , it's about 5% drop on predicting probability(quite a lot).
- Consider of missing value handling.
- Deleting rows which have missing values.

Naive Bayes classifier

```
[[73 74 71 78 72]
 [77 80 79 78 75]
 [74 74 78 83 80]
 [78 77 78 77 77]
 [75 86 81 78 82]
 [77 77 80 75 80]
 [78 73 73 75 80]
 [69 77 78 69 83]
 [78 79 74 79 82]
 [73 80 80 82 73]]
d_list length =50
max relative index = 21 diagonal_sum = 86
77.18
```

72.28%>>>77.18%

conclusion

- Missing value is a significant issue for data analyzing, can't just replace it with simple function.
- Can use entropy for reducing the amount of the data that have to be processed.

✔ Future work:

1.get more precise method for missing value.

2.compare more dataset and classify method.