WEEKLY REPORT 4

Paper Name:

Hard Drive Failure Prediction Using Classification and Regression Trees

Tasks:

1. Data Preprocessing:

Disk Model: Seagate A   2017 Jan-Dec

Difficulty: This paper sampled disk data every hour, but our dataset only has daily data, so I find it hard to achieve comparable level of accuracy

Statistics:

Paper:

Family	Class	Disks	Period	Samples
"W"	Good	22, 790	56 days	30, 631, 028
"W"	Failed	434	20 days	158, 190

My work:

Family	Class	Disks	Period	Samples
Seagate A	Good	34, 131	365 days	12,055,627
Seagate A	Failed	1058	365 days	182, 070

Comparing the above statistics, I find that I have **more failed disk samples** but **less good samples**. And the main difficulty is that my **time period is too long** to accurately predict a failure.

2. Feature selection:

12 basic features:

ID #	Attribute Name
1	**Raw Read Error Rate ???**
2	Spin Up Time
3	Reallocated Sectors Count
4	Seek Error Rate
5	Power On Hours
6	Reported Uncorrectable Errors
7	High Fly Writes
8	Temperature Celsius
9	**Hardware ECC Recovered**
10	**Current Pending Sector Count**
11	Reallocated Sectors Count (raw value)
12	**Current Pending Sector Count (raw value)**

Problem:

1) My dataset doesn't have data of the attribute '**Hardware ECC Recovered**'. So I used 11 features in total.

2) The paper says it uses **10 normalised value: 1-10 plus 2 raw values: 11-12**. However, the table above shows that it uses **'raw read error rate'.** I'm a bit confused here so I choose to use raw value here. But I observed that the raw value of this attribute is quite unstable and really fluctuates dramatically.

3) This paper uses mostly **normalised values** in dataset. But as the first paper I have worked on suggests that **the raw values** are far more informative in predicting failure instead. I summarised the standard deviations of each attribute on the whole dataset. Compared to raw values, the normalised values actually don't vary too much:

```
smart_1_raw               7.048288e+07
smart_3_normalized        2.521527e+00
smart_5_normalized        2.023904e-01
smart_7_normalized        4.185268e+00
smart_9_normalized        9.122558e+00
smart_187_normalized      9.315891e-01
```

```
smart_189_normalized    6.182678e+00
smart_194_normalized    4.475583e+00
smart_197_normalized    5.350651e-02
smart_5_raw             2.609522e+02
smart_197_raw           9.977192e+00
```

3. Train & test set split:

For each good drive:

Training data: randomly choose **3 samples** from the earlier 70%

Testing data: the later 30% samples

For each failed drive:

1) Divide them **randomly** into training and test sets in a 7 to 3 ratio
2) For training data, take out the failed sample within a ***time window***, that is, the last *n* hours before the failure actually occurs. For current stage, I choose n=12.

After preprocessing:

Training set: 102387 healthy samples + 8772 failed samples

Testing set: 102387 healthy samples + 3648 failed samples

4. Classification:

1) The first attempt: classification decision tree

CT parameters:

$MinimumSplit = 20, MinimumBucketSize = 7, ComplexityParameter = 0.001.$

FAR=49% (too high)

FDR=0.933 (close to 95.49%)

Problem analysis: A reasonable accuracy with too low precision often indicates the problem of **class imbalance**

2) The second attempt: **downsampling by randomly selection**

New training set: 102387 healthy samples-> 10236 healthy samples

Healthy/failed samples ratio = 1.17

New testing set: 102387 healthy samples-> 10236 healthy samples (the same set of disks as in training set)

Healthy/failed samples ratio = 2.8

Classification tree result:

FAR=34.5% (lower than previous, but still too high)

FDR=73.3% (too low)

3) Disk failure prediction:

Rule:

If any sample is classified as failed-> predict breakdown

Otherwise->a good drive

FAR=0.365 (still too high)

FDR=0.918

3)BP ANN:

Layer: 3 layer

Dense: 12, 20 and 1 nodes

maximum number of iterations=400

learning rate=0.1