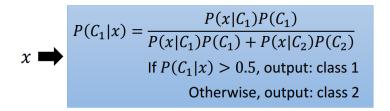
```
1.Logistic regression function.
vector<vector<double> > Data(59, vector<double>(4001));
4001 筆資料, 第1 維固定是1, 第2~58 維是 feature, 第59 維是 label
vector<double>W(parnum);
vector<double>G(parnum);
vector<double>D(parnum);
W、G、D 分別有 58 維·W 用來存 b+w*xi 中的 b 和 w·G 是 adagrad 中的參數·D 是微分
值。
for(int k=0;k<time;k++) //time 是總共 training 的次數
{
       vector<double>Y(datanum); //用來存每筆資料的估計值
       for(int n=0;n<datanum;n++)</pre>
       {
           for(int m=0;m<parnum;m++)</pre>
               Y[n]=Y[n]+W[m]*Data[m][n]; //計算每筆資料的 y=sigma(wxi+b)
           for(int m=0;m<parnum;m++)</pre>
           {
               D[m] = D[m] - 2*(Data[58][n] - (1/(1 + exp((-1)*Y[n]))))*Data[m][n];
                  //計算微分值並加上 sigmoid function
               G[m]=G[m]+D[m]*D[m];
                  //計算 adagrad 的 G
           }
       }
       for(int m=0;m<parnum;m++)</pre>
       {
           W[m]=W[m]-rate*D[m]/sqrt(G[m]); //更新 W 值
           D[m]=0; //將微分值歸 0
       }
}
```

2.Describe your another method, and which one is best.

我的第二個方法是 Naive Bayes Classifier 先假設每個 feature 的值都是高斯分布,而且每個 dimension 都是獨立的。假設每個 P(xi|C0)和 P(xi|C1)都是一維高斯分布。用 training set 計算 出 P(C0)、P(C1)、Gaussian 的 mean 57 個和 variance 57 個,當作 model。 在 testing set 算答案的時候將每一個 feature 帶入相對應的高斯然後求出機率後相乘 $P(x|C_1) = P(x_1|C_1) P(x_2|C_1)$ …… $P(x_k|C_1)$ …… $P(x_k|C_$



此外,為了方便,有先取 log 之後再比大小。

Training:

```
先把 training set 的 data 根據不同 label 分成兩半 D0 和 D1 · D0 有 num 筆 data
for(int i=0;i<57;i++) //分別計算 2 個 class 中 57 個 feature 的 mean
{
     for(int j=0;j<num;j++)
          m0[i]=m0[i]+D0[j][i];
     m0[i]=m0[i]/double(num);
     for(int j=0;j<(4001-num);j++)
          m1[i]=m1[i]+D1[j][i];
     m1[i]=m1[i]/double(4001-num);
}
for(int i=0;i<57;i++) //分別計算 2 個 class 中 57 個 feature 的 variance
{
     for(int j=0;j<num;j++)
          v0[i]=v0[i]+(D0[j][i]-m0[i])*(D0[j][i]-m0[i]);
     v0[i]=v0[i]/double(num);
     for(int j=0;j<(4001-num);j++)
          v1[i]=v1[i]+(D1[j][i]-m1[i])*(D1[j][i]-m1[i]);
     v1[i]=v1[i]/double(4001-num);
}
Testing:
for(int j=0; j<600; j++)
```

```
{
    double mul0=0,mul1=0; //用來計算機率相乘的結果.這裡改用 log 相加
    for(int i=0;i<57;i++)
    {
        if(abs(v0[i])>0.0000000001 && abs(v1[i])>0.0000000001) //確定 var!=0
        {
            mul0=mul0+Gaussian(m0[i],v0[i],DData[i][j]);
            mul1=mul1+Gaussian(m1[i],v1[i],DData[i][j]); //計算機率取 log 相加
        }
    }
    if(mul0+log(c0) > mul1+log(c1)) ans=0; //如果 P[C0]*P[x|C0]比較大.就是 class 0
    else ans=1;
    DData[57][j]=ans;
}
```

結果做出來是方法一比較好,正確率約可達到 0.93,方法二的正確率只有 0.83(在 training set 計算和由 kaggle 計算都差不多)。

Naive Bayes 比較簡單、比較快,只要稍微計算就可以有 model,效果也不會太差。training set 小的時候,high bias/low variance 的 Naive Bayes 應該會比較好,但他沒辦法做出很準確的 model。

Logistic Regression 需要 train 一些時間。training set 小的時候·low bias/high variance 的 logistic regression 應該會比較差,因為可能會 overfit。但他的 feature 之間可以有關連,會有 一個比較準確的 model。

覺得這次作業的 training set 應該是算大·所以用 logistic regression 可以做出比較好的 model 吧!此外用 Naive Bayes 有一些小問題,以下會描述。

3.Other discussion.

在做方法二的時候遇到幾個小問題:

Naive Bayes 在某個 class 的某個 feature 上的 mean 和 variance 都是 $0 \cdot$ 所以這一維的數據並沒辦法列入計算,可能是因為這樣所以有一些不好的效果。

另外·在做 naïve bayes 的過程中·程式有些小錯·結果做出來的結果居然比最後正確版的更好!? 推測可能是因為 naïve bayes 並沒有辦法好好地描述這個 model 吧!

討論的對象: b02502108 陶昇永