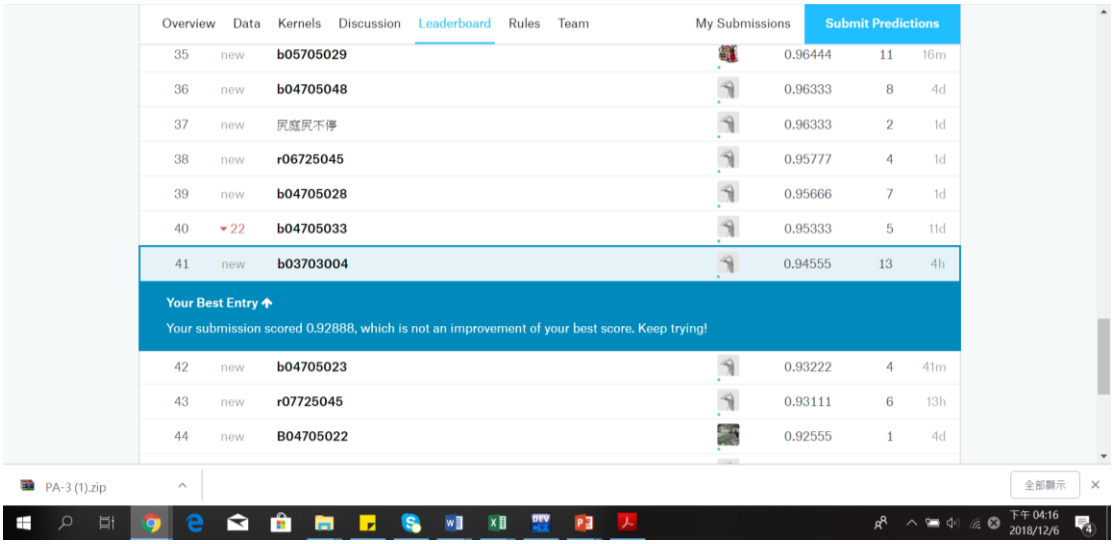
B03703004 財金五陳冠宇

Introduction to Information Retrieval HW03

執行環境：Dev C++ 5.11

使用語言：C++

結果：



參數選擇：

Feature<500

Chi-square > 20

MI>1.5

每篇總字數最多 45

P(t=1,c=1)>6 or P(t=1,c=0)=0

((P(t=1,c=1)- E[P(t=1,c=1)]) + (P(t=0,c=0)- E[P(t=0,c=0)]) - (P(t=0,c=1)- E[P(t=0,c=1)]) - (P(t=1,c=0)- E[P(t=1,c=0)]))>22

➔number of features 472, score = 0.92666

Feature>500

Chi-square > 0

MI>1.5

P(t=1,c=1)>1

➔number of features = 3342, score = 0.94555

各函數介紹：(1. Feature Selection　2. Train model　3. Apply model)

✓ Feature Selection

1. Chi-square
2. Pointwise MI
3. (在 class_i 內的 df 值>某數 || 在非 class_i 內的 df 值<某數)
4. Chi-square 表格(1,1)+(2,2)-(1,2)-(2,1)>某數
5. 每個 class 所擁有 feature 數<某數

※參數選擇請見第一頁

```
913  void SelectFeatures(int classes[][16])
914  {
915      map<string, double> dict1;
916      int total = 0;
917      for(int i=0; i<13; i++){
918          for(int j=1; j<16; j++){
919              string doc = std::to_string(classes[i][j]);
920              ifstream docM("C:\\Users\\Mark\\Desktop\\大五\\資訊檢索\\IRTM_OUT\\"+doc+".txt");
921          string line;
922          char *t, *s;
923          getline(docM, line);
924          char delim[] = " ";
925          while (getline(docM, line)){
926              t = strtok ((char*)line.c_str(), delim);   //parse with delim
927              while (t != NULL){
928                  string term(t);
929                  s = strtok(NULL, delim);   //t pointing to the next delimiter position
930                  if(dict1.find(term) == dict1.end())
931                  {
932                      dict1[term] = 1;
933                      //cout<<term<<endl;
934                  }
935                  else
936                      dict1[term] += 1;
937                  t = strtok(NULL, delim);
938              }
939          }
940      }
941      }
942      ofstream outfile ("C:\\Users\\Mark\\Desktop\\大五\\資訊檢索\\IRTM_Class\\merge.txt");
943      outfile<<std::left<<setw(70)<<"term"<<setw(70)<<"total_num"<<endl;
944      map<string, double>::iterator it;
945      for(it = dict1.begin(); it != dict1.end(); it++){
946          outfile<<std::left<<setw(70)<<(*it).first;
```

```
947          outfile<<std::left<<setw(70)<<(*it).second<<endl;
948      }
949
950      for(int i=0; i<13; i++){
951          map<string, Arr5> score;
952          for(int j=1; j<16; j++){
953              string doc = std::to_string(classes[i][j]);
954              ifstream docM("C:\\Users\\Mark\\Desktop\\大五\\資訊檢索\\IRTM_OUT\\"+doc+".txt");
955          string line;
956          char *t, *s;
957          getline(docM, line);
958          char delim[] = " ";
959          while (getline(docM, line)){
960              t = strtok ((char*)line.c_str(), delim);   //parse with delim
961              while (t != NULL){
962                  string term(t);
963                  s = strtok(NULL, delim);   //t pointing to the next delimiter position
964
965                  if(score.find(term) == score.end())
966                  {
967                      score[term] = {1,0,0,0,0};
968                  }
969                  else
970                      score[term].num[0]++;
971
972                  t = strtok(NULL, delim);
973              }
974          }
975      }
976      string now = std::to_string(i+1);
977      ofstream outfile ("C:\\Users\\Mark\\Desktop\\大五\\資訊檢索\\IRTM_Class\\fs_"+now+".txt");
978      map<string, double> D2V;
979      map<string, double> D2V2;
980      map<string, Arr5>::iterator it1;
```

```
981     for(it1 = score.begin(); it1 != score.end(); it1++){
982         double expected_f[4];
983         double chi_score = 0;
984         score[(*it1).first].num[1] = 15-score[(*it1).first].num[0];
985         score[(*it1).first].num[2] = dict1[(*it1).first] - score[(*it1).first].num[0];
986         score[(*it1).first].num[3] = 180 - score[(*it1).first].num[2];
987         expected_f[0] = 195.0*(15.0/195)*(dict1[(*it1).first]/195.0);
988         expected_f[1] = 195.0*(15.0/195)*((195.0-dict1[(*it1).first])/195.0);
989         expected_f[2] = 195.0*(180.0/195)*(dict1[(*it1).first]/195.0);
990         expected_f[3] = 195.0*(180.0/195)*((195.0-dict1[(*it1).first])/195.0);
991         chi_score = pow(score[(*it1).first].num[0]-expected_f[0], 2)/(expected_f[0]) + pow(score[(*it1).first].num[1]-expected_f[1], 2)/(expected_f[1]) +
992             pow(score[(*it1).first].num[2]-expected_f[2], 2)/(expected_f[2]) + pow(score[(*it1).first].num[3]-expected_f[3], 2)/(expected_f[3]);
993         double MI = log2((score[(*it1).first].num[0]/195.0)/((15.0/195)*(dict1[(*it1).first]/195)));
994         //MI = MI/(-log2((score[(*it1).first].num[0]/195.0)));
995         double EMI = score[(*it1).first].num[0]*log2(score[(*it1).first].num[0]/(15*dict1[(*it1).first]))
996         +score[(*it1).first].num[1]*log2(score[(*it1).first].num[1]/(15*(195.0-dict1[(*it1).first])))
997         +score[(*it1).first].num[2]*log2(score[(*it1).first].num[2]/(180.0*dict1[(*it1).first]))
998         +score[(*it1).first].num[3]*log2(score[(*it1).first].num[3]/(180.0*(195.0-dict1[(*it1).first])));
999         if((score[(*it1).first].num[0]>6||score[(*it1).first].num[2]<1) && MI>1.5 && ((score[(*it1).first].num[0]-expected_f[0])+(score[(*it1).first].nur
1000        (score[(*it1).first].num[1]-expected_f[1])-(score[(*it1).first].num[2]-expected_f[2]))>22)
1001            D2V[(*it1).first] = chi_score;
1002        else
1003            D2V[(*it1).first] = -chi_score;
1004        D2V2[(*it1).first] = MI;
1005        //outfile<<std::left<<setw(70)<<(*it1).first;
1006        /*outfile<<std::left<<setw(70)<<D2V[(*it1).first]<<"    "<<score[(*it1).first].num[0]<<"    "<<score[(*it1).first].num[1]
1007        <<"    "<<score[(*it1).first].num[2]<<"    "<<score[(*it1).first].num[3]<<"    "<<expected_f[0]<<"    "<<expected_f[1]
1008        <<"    "<<expected_f[2]<<"    "<<expected_f[3]<<"    "<<MI<<"    "<<EMI<<endl;*/
1009    }
1010
1011    //&& ((score[(*it1).first].num[1]-expected_f[1])+(score[(*it1).first].num[2]-expected_f[2]))<0 && (expected_f[0]>1 || chi_score * expected_f[0]>20)
1012    //&&((score[(*it1).first].num[0]-expected_f[0])+(score[(*it1).first].num[3]-expected_f[3]))>0 && ((score[(*it1).first].num[1]-expected_f[1])+(score[
1013    vector<pair<string,double> > vec (D2V.begin(), D2V.end());
1014    sort(vec.begin(),vec.end(),comp_by_value);
```

```
1015    vector<pair<string,double> >::iterator it5;
1016    int count = 0;
1017    for(it5 = vec.begin(); it5!= vec.end() && count<45; ++it5){
1018        if(D2V[(*it5).first]>20){
1019            // && (D2V2[(*it5).first] > 2) && count<50
1020            outfile<<std::left<<setw(70)<<(*it5).first;
1021            outfile<<std::left<<setw(70)<<D2V[(*it5).first]<<endl;
1022            count++;
1023            total++;
1024        }
1025
1026    }
1027    //printVec(vec);
1028    //cout<<endl<<endl;
1029    }
1030    cout<<total<<endl;
1031
1032    outfile.close();
1033 }
```

✓ Train model (Multinomial)

利用 13*15 個 Training documents 為每一個 class 計算 add-one smoothing 的 $P(X=t_k|c)$機率

```
732  void TrainMultinomialNB(int classes[][16], int num[13])
733  {
734      map<string, double> total;
735      for(int i=0; i<13; i++){
736          for(int j=1; j<16; j++){
737              string doc = std::to_string(classes[i][j]);
738              ifstream docM("C:\\Users\\Mark\\Desktop\\大五\\資訊檢索\\IRTM_OUT\\"+doc+".txt");
739              string line;
740              char *t, *s;
741              getline(docM, line);
742              char delim[] = " ";
743              while (getline(docM, line)){
744              t = strtok ((char*)line.c_str(), delim);   //parse with delim
745              while (t != NULL){
746                  string term(t);
747                  s = strtok(NULL, delim);   //t pointing to the next delimiter position
748                  double ans = atof(s);
749                  if(total.find(term) == total.end())
750                      total[term] = 1;
751                  t = strtok(NULL, delim);
752              }
753              }
754          }
755      }
756
757      for(int i=0; i<13; i++){
758          map<string, double> dict1;
759          int distinct_num = 0, total_num = 0;
760          string clas = std::to_string(i+1);
761          ofstream outfile ("C:\\Users\\Mark\\Desktop\\大五\\資訊檢索\\IRTM_Class\\"+clas+".txt");
762
763          map<string, double> featuredict;
764          string doc = std::to_string(i+1);
```

```
765          ifstream docM("C:\\Users\\Mark\\Desktop\\大五\\資訊檢索\\IRTM_CLASS\\fs_"+doc+".txt");
766          string line;
767          char *t, *s;
768          getline(docM, line);
769          char delim[] = " ";
770          while (getline(docM, line)){
771          t = strtok ((char*)line.c_str(), delim);   //parse with delim
772          while (t != NULL){
773              string term(t);
774              s = strtok(NULL, delim);   //t pointing to the next delimiter position
775              double ans = atof(s);
776              featuredict[term] = ans;
777              t = strtok(NULL, delim);
778          }
779          }
780
781
782          for(int j=1; j<16; j++){
783              string doc = std::to_string(classes[i][j]);
784              ifstream docM("C:\\Users\\Mark\\Desktop\\大五\\資訊檢索\\IRTM_OUT\\"+doc+".txt");
785              string line;
786              char *t, *s;
787              getline(docM, line);
788              char delim[] = " ";
789              while (getline(docM, line)){
790              t = strtok ((char*)line.c_str(), delim);   //parse with delim
791              while (t != NULL){
792                  string term(t);
793                  s = strtok(NULL, delim);   //t pointing to the next delimiter position
794                  double ans = atof(s);
795                  if(dict1.find(term) == dict1.end())
796                  {
797                      dict1[term] = ans;
798                      distinct_num++;
```

```
                    total_num++;
                }
                else
                {
                    dict1[term] += ans;
                    total_num++;
                }
                t = strtok(NULL, delim);
            }
            }
        }
        outfile<<std::left<<setw(70)<<"term"<<setw(70)<<"prob"<<endl;
        map<string, double>::iterator it;
        for(it = dict1.begin(); it != dict1.end(); it++){
            if(featuredict.find((*it).first) != featuredict.end())
            {
                outfile<<std::left<<setw(70)<<(*it).first;
                outfile<<std::left<<setw(70)<<((*it).second+1)/(total_num+total.size()-1)<<endl;
            }
        }
        outfile.close();
        num[i] = total_num+total.size()-1;
    }
}

```

✓ Apply model

利用各 testing documents 的 df dictionary 做 log 機率連加，若沒有出現在 training documents 中的 term 就直接略過，最大的即是分類答案

```
int ApplyMultinomialNB(string d, int num[13])
{
    int result = -9999999;
    int class_result = 0;
    map<string, int> dict1;

    ifstream docM("C:\\Users\\Mark\\Desktop\\大五\\資訊檢索\\IRTM_OUT\\"+d);
    string line;
    char *m, *n;
    getline(docM, line);
    char delim[] = " ";
    while (getline(docM, line)){
    m = strtok ((char*)line.c_str(), delim);   //parse with delim
    while (m != NULL){
        string term(m);
        n = strtok(NULL, delim);   //t pointing to the next delimiter position
        double ans = atoi(n);
        dict1[term] = ans;
        m = strtok(NULL, delim);
    }
    }

    map<string, double> dictT;
    for(int i=1; i<=13; i++)
    {
        double temp = 0;
        string doc = std::to_string(i);
        ifstream docM("C:\\Users\\Mark\\Desktop\\大五\\資訊檢索\\IRTM_Class\\"+doc+".txt");
        string line;
        char *t, *s;
        getline(docM, line);
        char delim[] = " ";
        while (getline(docM, line)){
        t = strtok ((char*)line.c_str(), delim);   //parse with delim
```

```
        while (t != NULL){
            string term(t);
            s = strtok(NULL, delim);   //t pointing to the next delimiter position
            double ans = atof(s);
            dictT[term] = ans;
            t = strtok(NULL, delim);
        }
        }
    }

    for(int i=1; i<=13; i++)
    {
        map<string, double> dict2;
        double temp = 0;
        string doc = std::to_string(i);
        ifstream docM("C:\\Users\\Mark\\Desktop\\大五\\資訊檢索\\IRTM_Class\\"+doc+".txt");
        string line;
        char *t, *s;
        getline(docM, line);
        char delim[] = " ";
        while (getline(docM, line)){
        t = strtok ((char*)line.c_str(), delim);   //parse with delim
        while (t != NULL){
            string term(t);
            s = strtok(NULL, delim);   //t pointing to the next delimiter position
            double ans = atof(s);
            dict2[term] = ans;
            t = strtok(NULL, delim);
        }
        }
        map<string, int>::iterator it;
        for(it = dict1.begin(); it != dict1.end(); it++){
            if(dict2.find((*it).first) == dict2.end() && (dictT.find((*it).first) != dict2.end()))
                temp  = temp +  (*it).second*log(1.0/num[i-1]);
```

```
            else if (dict2.find((*it).first) != dict2.end() )
                temp  = temp + (*it).second*log((dict2[(*it).first]));
            else;
            //cout<<"****"<<(*it).second<<" "<<(*it).first<<" "<<(pow((*it).second, (dict1[(*it).first])))<<"  "<<temp<<endl;
            /*if(d=="129.txt")
                cout<<(*it).first<<"   "<<temp<<endl;*/
        }
        //if(d=="127.txt")

        //cout<<temp*(1.0/13)<<" "<<result<<endl;
        if(temp*(1.0/13)>result)
        {
            result = temp*(1.0/13);
            class_result = i;
        }
        //cout<<"文章"<<d<<" "<<i<<" "<<result<<endl;
    }
    return class_result;

}
```

✓ 主程式

Feature Selection -> Train Multinomial Model -> Apply the model and return most possible class

```
    int classes[13][16];
    int testing[195];
    int num[13];
    int count = 0;
    char *kk;
    int class_num = 0;
    int p = 0;
    string line2;
    ifstream doc2("C:\\Users\\Mark\\Desktop\\大五\\資訊檢索\\class.txt");
    while (getline(doc2, line2)){
        p = 0;
        kk = strtok ((char*)line2.c_str(), " ");   //parse with delim
        while (kk != NULL){
            classes[class_num][p] =  atoi(kk);      //copy the string pointed to char array 'target'
            if(p!=0)
            {
                testing[count] = atoi(kk);
                count++;
            }
            //cout<<p<<" "<<(classes[class_num][p])<<" ";
            p++;
            //cout<<"count"<<" "<<count<<" ";
            kk = strtok(NULL, " ");   //t pointing to the next delimiter position
        }
        //cout<<endl;
        class_num++;
    }
    SelectFeatures(classes);
    TrainMultinomialNB(classes, num);

    char InputPathA[65535] = "C:\\Users\\Mark\\Desktop\\大五\\資訊檢索\\IRTM_OUT";    //放要讀取檔案的資料夾路徑到InputPath字串裡
    ofstream myfile ("C:\\Users\\Mark\\Desktop\\大五\\資訊檢索");
    myfile.open ("result.csv");
    myfile <<"Id"<<","<<"Value"<<"\n";
```

```
    char szDirA[65535];
    char dirA[65535];
    int startid = 0;
    WIN32_FIND_DATA AFileData;
    HANDLE          AhList;
    sprintf(szDirA, "%s\\*", InputPathA );
    int f=0;
    if ( (AhList = FindFirstFile(szDirA, &AFileData))==INVALID_HANDLE_VALUE )
        printf("No files be found.\n\n");
    else {
        while (1) {
            if (!FindNextFile(AhList, &AFileData)) {
                if (GetLastError() == ERROR_NO_MORE_FILES)
                    break;
            }
            else{
                int is_test = 0;
                f++;
                string x = AFileData.cFileName;
                for(int i=0; i<195; i++)
                {
                    if(atoi(x.substr(0, x.find(".txt", 0)).c_str())==testing[i])
                    {
                        is_test = 1;
                        break;
                    }
                }
                if (startid>0 && is_test!=1)
                {
                    myfile << x.substr(0, x.find(".txt", 0))<<","<<ApplyMultinomialNB(x, num)<<"\n";
                    }
                startid ++;
            }
        }
```

```
    }


    return 0;
}
```