Classification and Clustering for the Microarray Data Analysis

12171628 방솔찬

001분반 assignment

연락처: 010-7538-3823

이메일:bsc981111@naver.com

과제 개요: 주어진 생물 데이터 Classification and Clustering

개발 환경: colab, python 3.6.5, 'Linux-4.19.104+-x86_64-with-Ubuntu-18.04-bionic'

사용 라이브러리:

1. import numpy as np-> 불러온 data를 numpy array에 저장& sqrt & power
2. from collections import Counter->knn알고리즘에서 점 주변의 target개수 셀 때 사용
3. import os->system("pause") 실행

데이터: ribosom data (121*79)와 nonribo data(2346*79)의 data가 존재한다

데이터 경로: "/ribo-data.txt","/non-ribo-data.txt"

기능: **knn 알고리즘**과 **k-means 알고리즘을** 수행하여 각각 ribosom/nonribosom data classification,clustering 수행 후 6-fold algorithm수행하여 Sensitivity,Specificity,Accuracy로 평가한다.

Knn의 경우 파라미터로 k,p를 받는다 k의 경우 k개의 거리가 가까운 data의 target값을 확인하여 **그 비율이 p보다 높을경우 ribosom data로 판별하고 비율이 p보다 낮을 경우 nonribosom data로 판별한다.**

Kmc는 K=2로 설정되어있기 때문에 K값을 받을 필요는 없다. 주어진 data에 관해서 clustering이 2번 실행되는데 처음에는 ribo data와 non ribo data의 첫번째 값이 centroid initial 값이 된다 두번째 함수에서는 랜덤하게 두 점이 골라졌을 때에 관해서 실행이 된다.

K-fold cv의 경우 불러온 data를 shuffle 하고 6개로 나누고 각각 train 과 cv로 나눈 후 6번을 반복해서 활용하였다.

소스코드 사용법

knn: ribosom_data.txt 와 non ribosom_data.txt파일을 실행파일과 같은 경로에 위치해야지 실행파일이 정상적으로 작동한다. 설정후에는 콘솔창에 띄어지는대로 k,p값을 파라미터로 받고 knn알고리즘을 수행 후 k,p의 값과 TP,FP,TN,FN의 개수가 출력된다.그 후 **sensitivity,specitivity,accuracy**가 6-fold에서 계산되어 평균값이 출력된다.

K-means: ribosom_data.txt 와 non ribosom_data.txt파일을 실행파일과 같은 경로에 위치해야지 실행파일이 정상적으로 작동한다. 설정후 k=2로 설정되어 있기에 **k-means 알고리즘을 2번 수행한후 출력 값은 각 cluster별 들어있는 ribosom data의 gene number가 출력이 된다.**

이 때 각각 처음 k-means 알고리즘의 경우에는 처음에는 nonribo 와 ribo 의 데이터를 각각 하나씩 뽑았을 경우의 함수가 실행되고 그 다음에는 랜덤으로 데이터를 k 개 뽑았을 경우의 함수가 실행된다.

Part B

Varying K in K Nearest Neighbors With p=50%, report the cross-validation accuracy for the following values of K (accuracy =(TP+TN)/total):

1. K=1: 0.9566274827725983
2. K=5: 0.9716254560194568
3. K=20: 0.9720308066477503
4. K=50: 0.9700040535062829
5. K=100: 0.9700040535062829

How is the performance affected by different values of K?

6. Answer: **크게 관계없다.**

Are there any non-ribosomal genes that are consistently misclassified as ribosomal? If yes, list the gene numbers.

7. Answer: p=0.5일때 K가 10만넘어도 FN의 개수가 0개가 된다.

Part C

Using the value of K that gave you the best performance from part b, try the following different values of p and report the sensitivity and specificity your classifier obtains from 6-fold cross validation:

8. P=5%:

   1. Sensitivity = 0.9256198347107438 2. Specificity =0.6231884057971014

9. P=25%:

   1. Sensitivity =0.7355371900826446 2. Specificity =0.9957374254049446

10. p=50%:

    1. Sensitivity = 0.5537190082644629 2. Specificity =0.9987212276214834

11. p=75%:

    1. Sensitivity = 0.256198347107438 2. Specificity = 1.0

12. p=90%:

    1. Sensitivity = 0.1322314049586777 2. Specificity = 0.9995737425404945

13. p=100%:

    1. Sensitivity = 0.0743801652892562 2. Specificity = 1.0

14. What general trend does sensitivity follow with increasing p? (Choose all that apply)

    Answer: **A(decreasing)**

15. What general trend does specificity follow with increasing p? (Choose all that apply)

    Answer: **C(increasing)**

16. When might you be more interested in having high sensitivity? (Choose all that apply)

    Answer: **B,C**

17. When might you be more interested in having high specificity? (Choose all that apply)

    Answer: **A,D**

K-means clustering with microarray data. 18. K=2. Pick the first data point in both ribo.txt and nonribo.txt as your starting centers. Are all the ribosomal genes in the same cluster?

18. K=2. Pick the first data point in both ribo.txt and nonribo.txt as your starting centers. Are all the ribosomal genes in the same cluster?

Answer: **NO**

19. If your answer to the previous question is no, list all the ribosomal genes that are in the cluster that is different from the majority of the ribosomal genes

Answer: **Number 121 (RPL31B PROTEIN SYNTHESIS , RIBOSOMAL PROTEIN L31B)**

20. What percentage of genes in each cluster are ribosomal genes?

Answer: **0.06 21.66**

21. K=2, choose two random data point as your starting centers.

Answer: **0.18, 0**

22. Comparing your results from choosing the first data point as the starting centers with those from choosing two random data point as the starting centers, are the clustering assignments for each gene the same?

Answer: **NO**

23. What can you say about K-means clustering based on the question 22?

Answer: 내가 임의로 2개를 고를 때와 랜덤선택 할 때의 cluster별 유전자 개수도 각각 554,1913개와 642,1825개로 달라진다. 또한 각각 클러스터별 ribosom data의 개수도 임의로 2개고를때는 언제나 121번 유전자가 오분류 되지만 랜덤하게 고를때는 121번 유전자가 오분류될때도 존재하지만 정확하게 분류 될 때도 존재한다.

24. Do K-means clustering on the same dataset for 20 times with K=2 and random starting centers. Are there any ribosomal genes that are often clustered into a different cluster from the majority of the ribosomal genes?

Answer: **Yes**

25. If there are not, you can skip this question. If there are, specify their index numbers.
Answer: **Number 121 (RPL31B PROTEIN SYNTHESIS , RIBOSOMAL PROTEIN L31B)**

Knn Pseudocode:

```
def cross_val_score():
    get k and p # k와 p값 입력 받기
    TP=0,FP=0,TN=0,FN=0 # 값 초기화
    Divide len(data) into 6 # 6-fold 실행
    for I in range(0,6):# 6-fold 실행
        points<- training data
        for j in c_v data
            find_nearest_neighbors(c_v data[j],points,k,p)
    calculate accuracy sensitivity specificity

def find_nearest_neighbors(c_v data[],points,k,p):
    store distance between training datas and c_v data
    sort the distance and choose the short k among them
    if the ratio of ribosoms in the list judge ribosom
```

else judge non ribosom

def distance(p1,p2):

return sqrt(sum(p2-p2)^2))

K-means Pseudocode:

def random_selection(k):

shuffle and select k data

def select_each_ribo_noribo(k=2):

select first ribo data and first non ribo data

def k_means_select_each():

get k

centroid=select_each_nonribo(k)

centroids_old.shape=centroids.shape

labels.shape=len(data).shape

1

lable the cluster with a minimum distance by obtaining the given point from the k cluster points

the average of the labelled data is used to obtain the centroid.

If the centroid changed, run again from 1.

def k_means_select_random():

get k

centroid=random selection(k)

centroids_old.shape=centroids.shape

labels.shape=len(data).shape

1

lable the cluster with a minimum distance by obtaining the given point from the k cluster points

the average of the labelled data is used to obtain the centroid.

If the centroid changed, run again from 1.

K=20이고 p=0.1일때 knn 실행화면

```
1 print("knn 함수 실행...")
2 cross_val_score()
```

```
knn 함수 실행...
K값을 입력하세요 : 20
P값을 입력하세요 : 0.1


K=  20  P=  0.1
TP의 개수: 95
FP의 개수: 288
TN의 개수: 2058
FN의 개수: 26
sensitivity:  0.7851239669421488
specificity:  0.8772378516624041
Accuracy:  0.8727199027158492
```

K=2일때 k-means 실행화면

```
print("nonribo와 ribo data를 하나씩 뽑았을때의 k means 함수 실행!!")
k_means_select_each()
```

```
nonribo와 ribo data를 하나씩 뽑았을때의 k means 함수 실행!!
K=2
RIBO genes: gene number 0 to 120
NONRIBO genes: gene number 121 to 2466

Cluster  0 의 total number of genes:  554 이다
Cluster  0 의 ribodata number of genes:  120 이다

Cluster  1 의 total number of genes:  1913 이다
Cluster  1 의 ribodata number of genes:  1 이다

Cluster 0  :
index: 1 Gene number=  0
index: 2 Gene number=  1
index: 3 Gene number=  2
index: 4 Gene number=  3
index: 5 Gene number=  4
index: 6 Gene number=  5
index: 7 Gene number=  6
index: 8 Gene number=  7
index: 9 Gene number=  8
index: 10 Gene number=  9
index: 11 Gene number=  10
index: 12 Gene number=  11
index: 13 Gene number=  12
index: 14 Gene number=  13
index: 15 Gene number=  14
index: 16 Gene number=  15
index: 17 Gene number=  16
index: 18 Gene number=  17
index: 19 Gene number=  18
index: 20 Gene number=  19
index: 21 Gene number=  20
index: 22 Gene number=  21
index: 23 Gene number=  22
index: 24 Gene number=  23
index: 25 Gene number=  24
index: 26 Gene number=  25
```

index: 27 Gene number= 26
index: 28 Gene number= 27
index: 29 Gene number= 28
index: 30 Gene number= 29
index: 31 Gene number= 30
index: 32 Gene number= 31
index: 33 Gene number= 32
index: 34 Gene number= 33
index: 35 Gene number= 34
index: 36 Gene number= 35
index: 37 Gene number= 36
index: 38 Gene number= 37
index: 39 Gene number= 38
index: 40 Gene number= 39
index: 41 Gene number= 40
index: 42 Gene number= 41
index: 43 Gene number= 42
index: 44 Gene number= 43
index: 45 Gene number= 44
index: 46 Gene number= 45
index: 47 Gene number= 46
index: 48 Gene number= 47
index: 49 Gene number= 48
index: 50 Gene number= 49
index: 51 Gene number= 50
index: 52 Gene number= 51
index: 53 Gene number= 52
index: 54 Gene number= 53
index: 55 Gene number= 54
index: 56 Gene number= 55
index: 57 Gene number= 56
index: 58 Gene number= 57
index: 59 Gene number= 58
index: 60 Gene number= 59
index: 61 Gene number= 60
index: 62 Gene number= 61
index: 63 Gene number= 62
index: 64 Gene number= 63
index: 65 Gene number= 64
index: 66 Gene number= 65
index: 67 Gene number= 66
index: 68 Gene number= 67
index: 69 Gene number= 68
index: 70 Gene number= 69
index: 71 Gene number= 70
index: 72 Gene number= 71
index: 73 Gene number= 72
index: 74 Gene number= 73
index: 75 Gene number= 74
index: 76 Gene number= 75
index: 77 Gene number= 76
index: 78 Gene number= 77
index: 79 Gene number= 78
index: 80 Gene number= 79
index: 81 Gene number= 80
index: 82 Gene number= 81
index: 83 Gene number= 82
index: 84 Gene number= 83
index: 85 Gene number= 84
index: 86 Gene number= 85
index: 87 Gene number= 86
index: 88 Gene number= 87
index: 89 Gene number= 88
index: 90 Gene number= 89
index: 91 Gene number= 90
index: 92 Gene number= 91
index: 93 Gene number= 92
index: 94 Gene number= 93
index: 95 Gene number= 94
index: 96 Gene number= 95
index: 97 Gene number= 96
index: 98 Gene number= 97
index: 99 Gene number= 98
index: 100 Gene number= 99
index: 101 Gene number= 100
index: 102 Gene number= 101
index: 103 Gene number= 102
index: 104 Gene number= 103
index: 105 Gene number= 104
index: 106 Gene number= 105

```
index: 107 Gene number=  106
index: 108 Gene number=  107
index: 109 Gene number=  108
index: 110 Gene number=  109
index: 111 Gene number=  110
index: 112 Gene number=  111
index: 113 Gene number=  112
index: 114 Gene number=  113
index: 115 Gene number=  114
index: 116 Gene number=  115
index: 117 Gene number=  116
index: 118 Gene number=  117
index: 119 Gene number=  118
index: 120 Gene number=  119

Cluster 1  :
index: 1 Gene number=  120
```