## 聚类结果

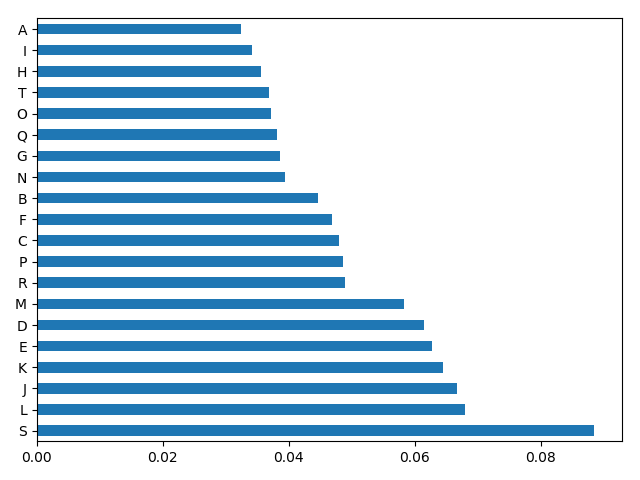
使用了不用聚类算法对Vt、Vs、Vts和skipgram\_KGofGPS\_tra矩阵进行聚类分析，结果相差不大，因此主要根据kmeans算法对聚类结果进行分析。

* Vt
  1. 聚类中心

见附件 output/outputVt4\_\_basic.csv

* 1. 特征重要性

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| A | B | C | D | E | F | G | H | I | J |
| 0.0324825 | 0.0446471 | 0.04792085 | 0.06152611 | 0.0627351 | 0.04688062 | 0.03862217 | 0.03567583 | 0.0342422 | 0.06668447 |
| K | L | M | N | O | P | Q | R | S | T |
| 0.0644733 | 0.06805946 | 0.05833332 | 0.03940019 | 0.0372056 | 0.04862565 | 0.03811306 | 0.04901552 | 0.08842917 | 0.03692779 |

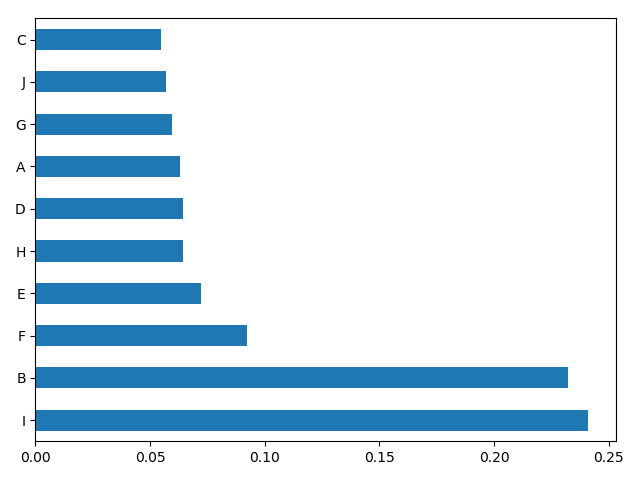


* Vts
  1. 聚类中心

见附件 output/outputVts4\_\_basic.csv

* 1. 特征重要性

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| A | B | C | D | E | F | G | H | I | J |
| 0.06311167 | 0.23234636 | 0.05454281 | 0.06421936 | 0.07200186 | 0.0921443 | 0.05933599 | 0.06434759 | 0.24101957 | 0.05693048 |



* Vts\_select
  1. 聚类中心

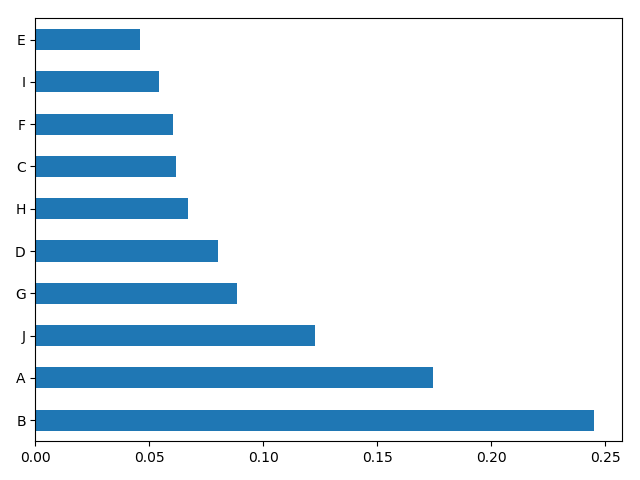
见附件 output/outputVts\_select4\_basic.csv

* skipgram\_KGofGPS\_tra
  1. 聚类中心

见附件 output/outputSkip4\_\_basic.csv

* 1. 特征重要性

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| A | B | C | D | E | F | G | H | I | J |
| 0.17441062 | 0.24495272 | 0.06173752 | 0.08016433 | 0.04577251 | 0.06052473 | 0.08844964 | 0.06693039 | 0.0543943 | 0.12266324 |



* VtVs
  1. 聚类中心

见附件 output/outputVtVs4\_\_basic.csv

* 1. 特征重要性

from A-AN：

[0.08468919 0.02643161 0.03834488 0.05829123 0.06165758 0.02750386

0.04730157 0.03651945 0.04857426 0.03864361 0.12797271 0.02715804

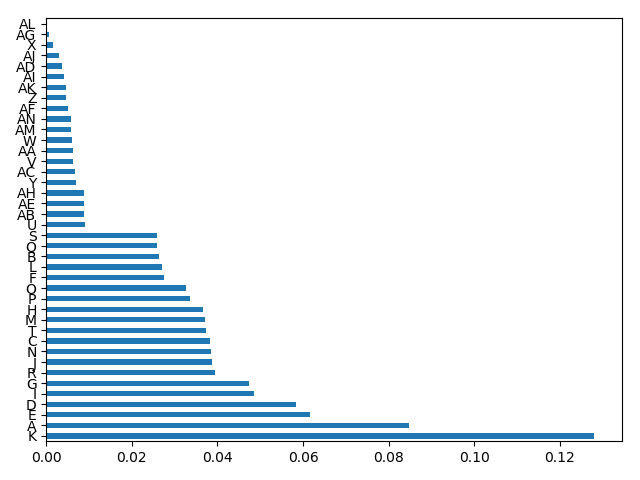
0.03709405 0.0384939 0.03276296 0.03361509 0.0259382 0.03932208

0.02584203 0.0372078 0.00898542 0.0062627 0.00598868 0.00151087

0.0070427 0.00464311 0.00624882 0.00888952 0.00677136 0.00358955

0.00885291 0.00500011 0.00069361 0.00880637 0.00405468 0.00300655

0.00457803 0.00018328 0.00581515 0.00571243]



* VtVsSkip
  1. 聚类中心

见附件 output/outputVtVsSkip4\_\_basic.csv

* 1. 特征重要性

from A-AX：

[3.78844950e-02 2.06203518e-02 4.31613697e-02 1.81542275e-02

1.96327745e-02 1.77966200e-02 1.91912900e-02 2.04676537e-02

2.19671343e-02 1.82725473e-02 1.78000786e-02 2.50290521e-02

1.72252595e-02 2.65394127e-02 4.23500516e-02 1.94723558e-02

1.57273452e-02 1.68621971e-02 8.05965697e-02 1.99717260e-02

5.82423485e-03 4.16432807e-03 4.64563342e-03 1.03564073e-03

3.79064023e-03 2.45420992e-03 4.86481375e-03 6.60622220e-03

4.59383314e-03 2.97861347e-03 4.31795876e-03 3.44956131e-03

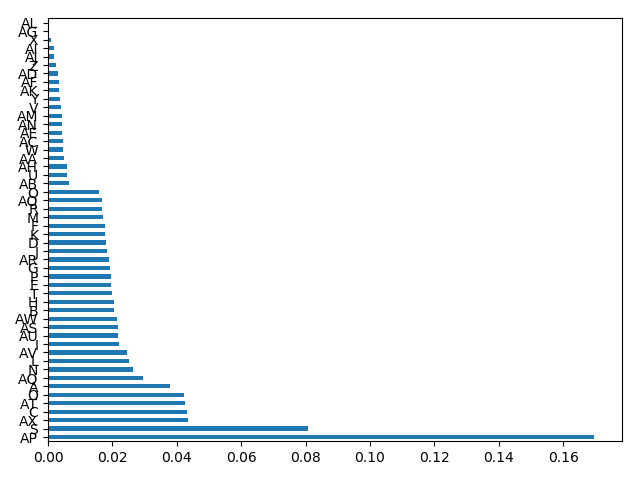
2.88361253e-04 5.78275819e-03 1.76416250e-03 2.01018691e-03

3.52742600e-03 4.12986977e-05 4.26385987e-03 4.27503208e-03

1.68258835e-02 1.69621576e-01 2.96568862e-02 1.88765375e-02

2.15981709e-02 4.26898591e-02 2.18304372e-02 2.46327531e-02

2.14471509e-02 4.34194580e-02]



* VtVsVts
  1. 聚类中心

见附件 output/outputVtVsVts4\_\_basic.csv

* 1. 特征重要性

from A-AX：

[0.02989234 0.02047107 0.0430389 0.0244036 0.02106596 0.02064794

0.02558576 0.01674815 0.01991518 0.02074386 0.01681682 0.0251498

0.06673678 0.01808957 0.02179955 0.02522575 0.02721837 0.02200048

0.02055984 0.01945189 0.00703881 0.00522523 0.00441446 0.00082628

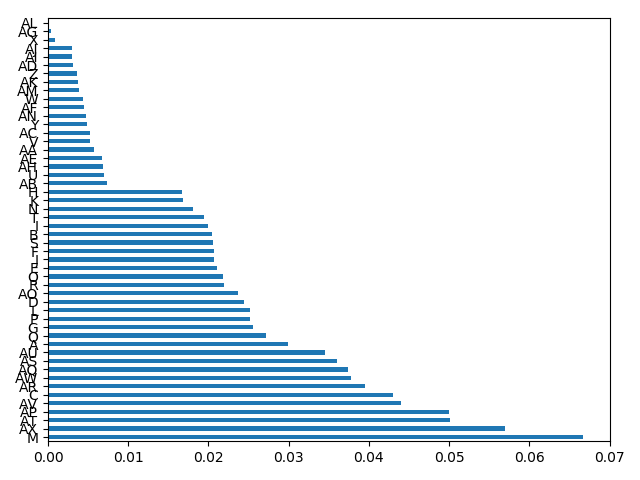
0.00487075 0.00364407 0.00573633 0.00739654 0.00521553 0.00309721

0.00672261 0.0045031 0.00037642 0.00682595 0.00301056 0.00295025

0.00372571 0.00017373 0.00391837 0.00477811 0.02371748 0.0499323

0.03740436 0.03956113 0.03604201 0.05010445 0.03446886 0.04402706

0.03777939 0.05695131]



## 聚类可视化

我们对于不同的聚类算法进行尝试，不同聚类算法的差别不是很大。可以根据簇内稠密度和簇外分散度，使用轮廓系数和CH来衡量。

* **轮廓系数**

向量与簇内部各点距离求均值，衡量簇内部的紧凑程度，再与簇外部所有点的距离求均值，衡量簇外部的分散程度。在Sklearn中是用silhouette\_score()计算所有点的平均轮廓系数，而silhouette\_samples()返回每个点的轮廓系数。

* **Calinski-Harabaz（CH）**

表示簇间的协方差矩阵与簇内的协方差矩阵相除，如果前者越大，后者越小，那么分值越大，分类越好，在sklearn中et是是用metrics.calinski\_harabaz\_score。

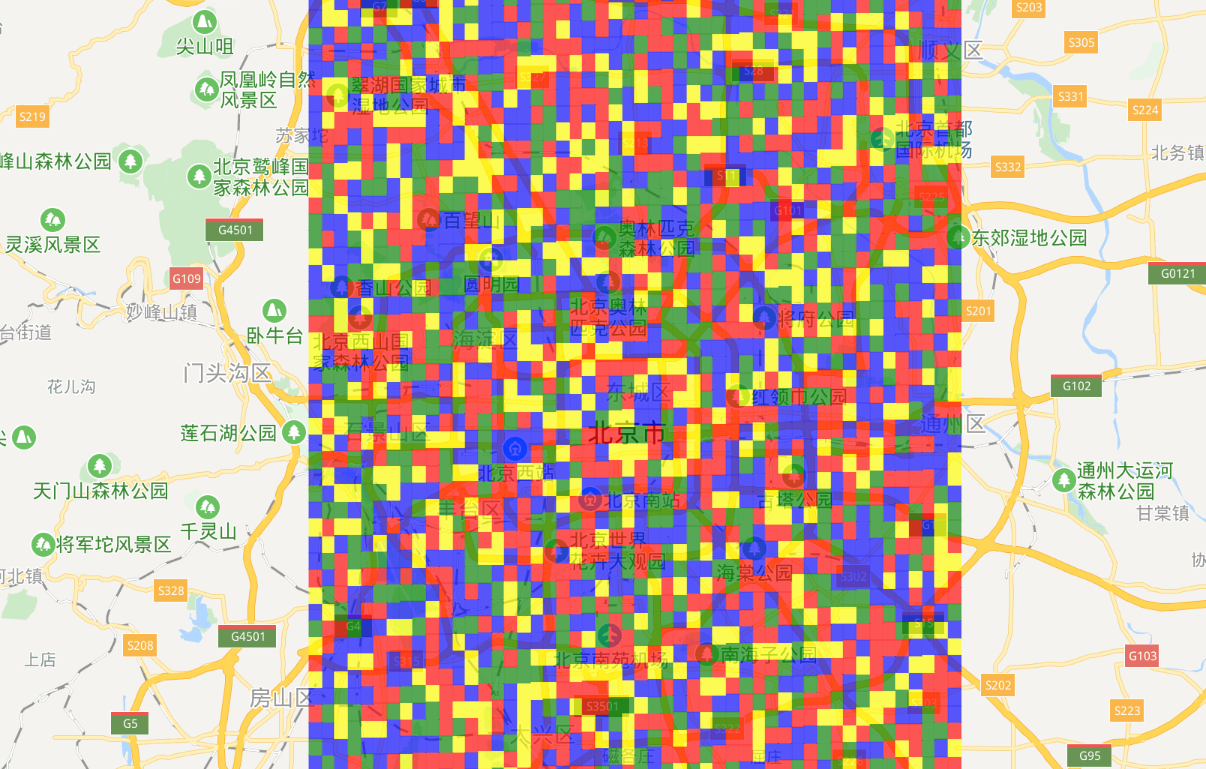
|  |  |  |
| --- | --- | --- |
| 聚类名称 | 轮廓系数 | Calinski-Harabaz |
| Kmeans | 78.15 | 0.03 |
| MiniBatchKMeans | 69.40 | 0.03 |
| Ward | 33.12 | 0.00 |
| AgglomerativeClustering | 4.30 | -0.08 |
| Birch | 1.52 | 0.02 |

选择轮廓系数和CH较大的方法，也就是KMeans进行下面的操作。

对于Vt,Vs,Vts, skipgram矩阵进行可视化处理。发现Vs难以进行聚类，结果往往是一个巨大的簇和只有零星几个点的小簇。故只对Vt, Vts, skipgram矩阵进行可视化处理。

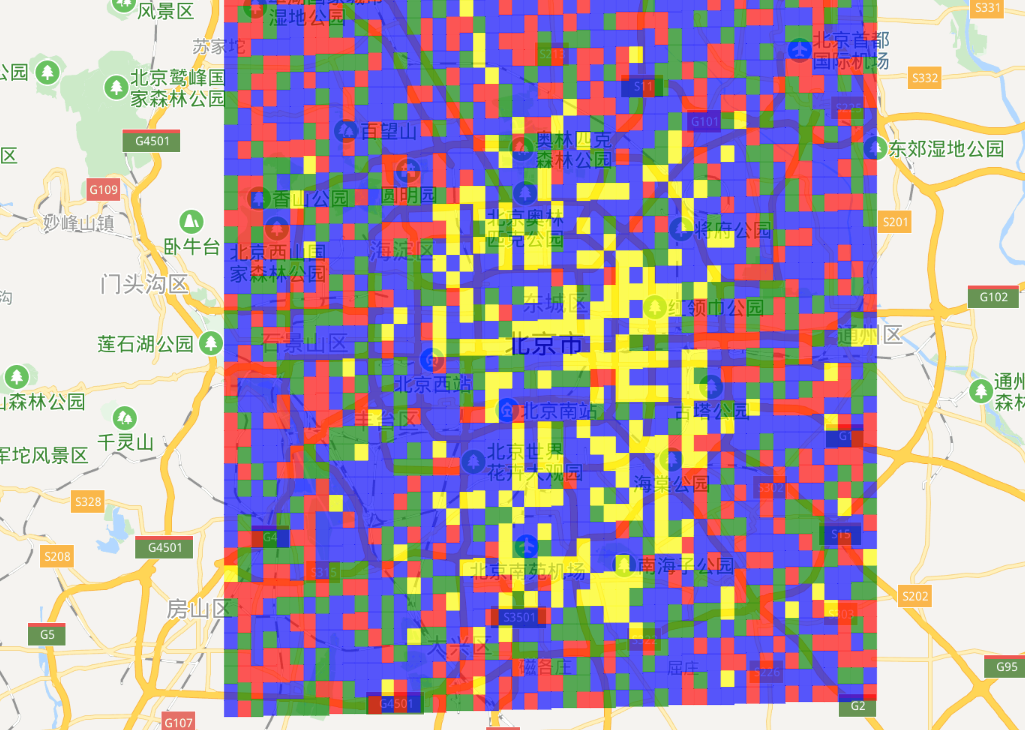
* **Vt**

文件Vt.html.



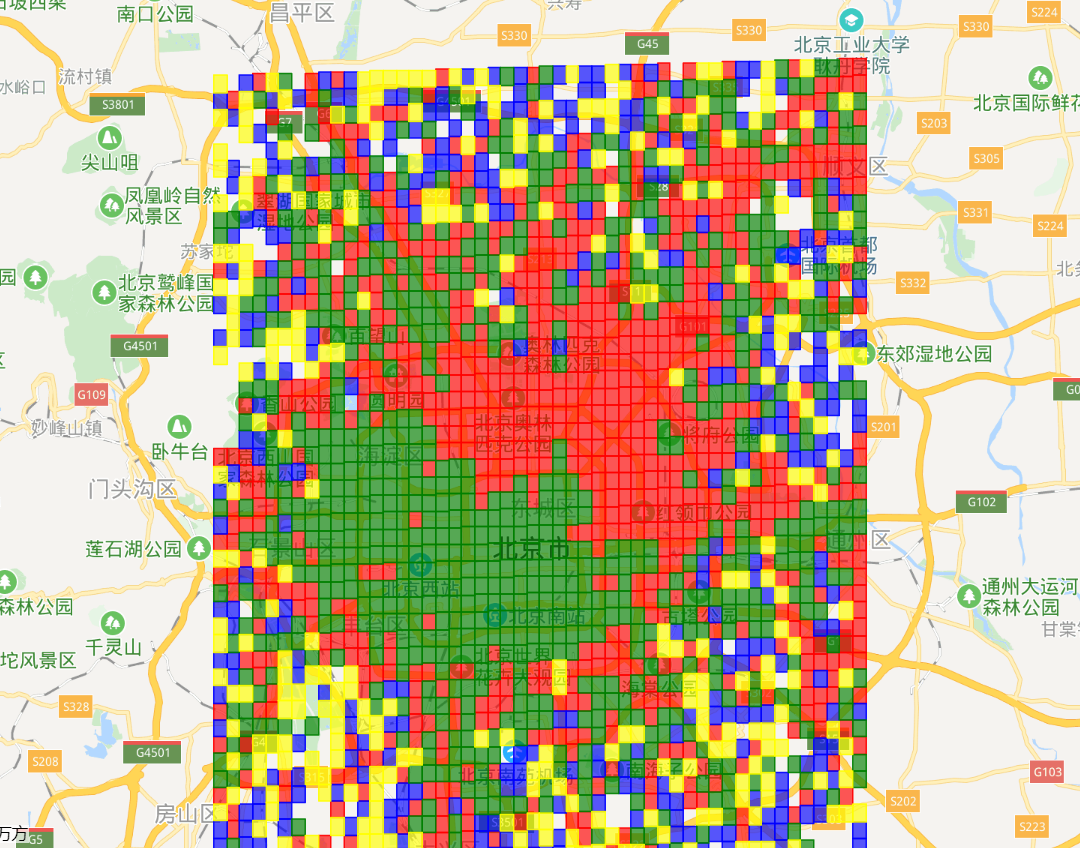
* **Vts**

文件Vts.html.



* **sk**

文件sk.html.



* **可解释性**

如果使用KMeans算法，可以查看变量的重要性对于聚类结果进行解释。也可以使用可解释算法。但是由于我们可视化的矩阵的列的意义并不明确，使用可解释的聚类算法的意义不大。所以我们仅使用KMeans并查看聚类中心和分类。