بسمه تعالی

دانشکده مهندسی کامپیوتر، دانشگاه صنعتی شریف

- مشخصات : عطیه جمشیدپور
- شماره دانشجویی:۹۴۱۰۳۸۳۵

گزارش فاز دوم پروژه بازیابی اطلاعات

موعد تحویل :۰۲/۲۵

به نام او

انتقال اسناد به فضای tf idf

برای این منظور کلاس tf idfرا پیاده سازی کردم که در ورودی خود نحوه بررسی اسنادرا به عنوان ورودی می‌گیرد. این ورودی می‌تواند در حالت‌های *None, lemmatization, stemming, stopword_removal* باشد. که بسته به هرکدام اعمال مربوطه‌را روی لغات انجام می‌دهد.

لازم به ذکر است که کد مربوط به این بخش در ضمیمه این سند در *phase2/tf_idf.py* آمده است.

(1) دسته‌بندی‌ها

1.1 دسته‌بند KNN

برای پیاده‌سازی این دسته بند کلاس KNN classifierرا پیاده سازی کردم. این دسته‌بند در ورودی خود distance modeرا می‌گیرد که می‌تواند دو مقدار cosine_similarity و euclidean_distance باشد. که متناسب با هر کدام فاصله‌را محاسبه می‌کند.

باتوجه به قدرت پردازش کم منابع سخت افزاری بنده، داده‌هارا به ۱/۱۰ تقلیل دادم. کد مربوط به این بخش در phase2/knn.py آمده است.

برای حالتی که هیچ پیش‌پردازش متنی نداشته باشیم و فاصله گذاری برحسب کسینوس‌زاویه بین بردارها باشد خروجی مطابق زیر است.

Output

Building tf idf table...

tf idf table built!

INFO:KNN:On Validation Data

INFO:KNN:K = 1:

Accuracy = 0.280

Recall_per class = [0.7727272727272727, 0.0, 0.02702702702702703, 0.2413793103448276]

Precision_per class = [0.2809917355371901, 0.0, 1.0, 0.2692307692307692]

macro_F1 = 0.180

confusion matrix = [[34. 0. 0. 10.]

[36. 0. 0. 4.]

[29. 2. 1. 5.]

[22. 0. 0. 7.]]

INFO:KNN:On Validation Data

INFO:KNN:K = 3:

Accuracy = 0.340

Recall_per class = [0.5454545454545454, 0.575, 0.0, 0.13793103448275862]

Precision_per class = [0.3, 0.38333333333333336, 0, 0.4]

macro_F1 = 0.263

confusion matrix = [[24. 15. 0. 5.]

[17. 23. 0. 0.]

[26. 10. 0. 1.]

[13. 12. 0. 4.]]

INFO:KNN:On Validation Data

INFO:KNN:K = 5:

Accuracy = 0.367

Recall_per class = [0.6363636363636364, 0.55, 0.02702702702702703, 0.13793103448275862]

Precision_per class = [0.358974358974359, 0.3384615384615385, 1.0, 0.6666666666666666]

macro_F1 = 0.290

confusion matrix = [[28. 15. 0. 1.]

[18. 22. 0. 0.]

[22. 13. 1. 1.]

[10. 15. 0. 4.]]

INFO:KNN:Best value for k: 5

INFO:KNN:On Train Data

INFO:KNN:K = 5:

Accuracy = 0.297

Recall_per class = [0.45964912280701753, 0.5986622073578596, 0.026936026936026935, 0.11912225705329153]

Precision_per class = [0.26844262295081966, 0.29200652528548127, 0.6153846153846154, 0.4418604651162791]

macro_F1 = 0.243

confusion matrix = [[131. 136. 2. 16.]

[109. 179. 0. 11.]

[138. 130. 8. 21.]

[110. 168. 3. 38.]]

INFO:KNN:On Validation Data

INFO:KNN:K = 5:

Accuracy = 0.367

Recall_per class = [0.6363636363636364, 0.55, 0.02702702702702703, 0.13793103448275862]

Precision_per class = [0.358974358974359, 0.3384615384615385, 1.0, 0.6666666666666666]

macro_F1 = 0.290

confusion matrix = [[28. 15. 0. 1.]

[18. 22. 0. 0.]

[22. 13. 1. 1.]

[10. 15. 0. 4.]]

برای حالتی که پیش پردازش متنی برای حذف stopword ها داشته باشیم و فاصله گذاری بر حسب کسینوس‌زاویه بین بردارها باشد خروجی مطابق زیر است.

Output

Building tf idf table...

tf idf table built!

INFO:KNN:On Validation Data

INFO:KNN:K = 1:

Accuracy = 0.393

Recall_per class = [0.36363636363636365, 0.375, 0.4594594594594595, 0.3793103448275862]

Precision_per class = [0.5, 0.39473684210526316, 0.3333333333333333, 0.3793103448275862]

macro_F1 = 0.393

confusion matrix = [[16. 14. 11. 3.]

[7. 15. 13. 5.]

[5. 5. 17. 10.]

[4. 4. 10. 11.]]

INFO:KNN:On Validation Data

INFO:KNN:K = 3:

Accuracy = 0.553

Recall_per class = [0.5227272727272727, 0.675, 0.4594594594594595, 0.5517241379310345]

Precision_per class = [0.6571428571428571, 0.6, 0.5666666666666667, 0.4]

macro_F1 = 0.547

confusion matrix = [[23. 8. 4. 9.]

[3. 27. 5. 5.]

[6. 4. 17. 10.]

[3. 6. 4. 16.]]

INFO:KNN:On Validation Data

INFO:KNN:K = 5:

Accuracy = 0.547

Recall_per class = [0.5, 0.675, 0.4594594594594595, 0.5517241379310345]

Precision_per class = [0.7586206896551724, 0.5, 0.5151515151515151, 0.47058823529411764]

macro_F1 = 0.543

confusion matrix = [[22. 13. 6. 3.]

[3. 27. 7. 3.]

[2. 6. 17. 12.]

[2. 8. 3. 16.]]

INFO:KNN:Best value for k: 3

INFO:KNN:On Train Data

INFO:KNN:K = 3:

Accuracy = 0.703

Recall_per class = [0.7263157894736842, 0.802675585284281, 0.6127946127946128, 0.6739811912225705]

Precision_per class = [0.680921052631579, 0.7339449541284404, 0.708171206225681, 0.6891025641025641]

macro_F1 = 0.702

confusion matrix = [[207. 28. 26. 24.]

[24. 240. 18. 17.]

[30. 29. 182. 56.]

[43. 30. 31. 215.]]

INFO:KNN:On Validation Data

INFO:KNN:K = 3:

Accuracy = 0.553

Recall_per class = [0.5227272727272727, 0.675, 0.4594594594594595, 0.5517241379310345]

Precision_per class = [0.6571428571428571, 0.6, 0.5666666666666667, 0.4]

macro_F1 = 0.547

confusion matrix = [[23. 8. 4. 9.]

[3. 27. 5. 5.]

[6. 4. 17. 10.]

[3. 6. 4. 16.]]

به نظر می‌آید پیش پردازش برای حذف *stopwords* ها به دقت مدل کمک می‌کند.

برای حالتی که هیچ پیش‌پردازش متنی نداشته باشیم و فاصله گذاری برحسب فاصله اقلیدسی بین بردارها باشد خروجی مطابق زیر است.

Output

Building tf idf table...

tf idf table built!

INFO:KNN:On Validation Data

INFO:KNN:K = 1:

Accuracy = 0.380

Recall_per class = [0.36363636363636365, 0.4, 0.2702702702702703, 0.5172413793103449]

Precision_per class = [0.41025641025641024, 0.43243243243243246, 0.4, 0.30612244897959184]

macro_F1 = 0.377

confusion matrix = [[16. 7. 7. 14.]

[11. 16. 4. 9.]

[6. 10. 10. 11.]

[6. 4. 4. 15.]]

INFO:KNN:On Validation Data

INFO:KNN:K = 3:

Accuracy = 0.440

Recall_per class = [0.5454545454545454, 0.5, 0.24324324324324326, 0.4482758620689655]

Precision_per class = [0.36923076923076925, 0.4878048780487805, 0.6, 0.4482758620689655]

macro_F1 = 0.432

confusion matrix = [[24. 8. 4. 8.]

[16. 20. 0. 4.]

[13. 11. 9. 4.]

[12. 2. 2. 13.]]

INFO:KNN:On Validation Data

INFO:KNN:K = 5:

Accuracy = 0.500

Recall_per class = [0.5454545454545454, 0.65, 0.2972972972972973, 0.4827586206896552]

Precision_per class = [0.4897959183673469, 0.5416666666666666, 0.55, 0.42424242424242425]

macro_F1 = 0.486

confusion matrix = [[24. 6. 5. 9.]

[11. 26. 0. 3.]

[6. 13. 11. 7.]

[8. 3. 4. 14.]]

INFO:KNN:Best value for k: 5

INFO:KNN:On Train Data

INFO:KNN:K = 5:

Accuracy = 0.637

Recall_per class = [0.7929824561403509, 0.6989966555183946, 0.5151515151515151, 0.5517241379310345]

Precision_per class = [0.5635910224438903, 0.6005747126436781, 0.7611940298507462, 0.704]

macro_F1 = 0.635

confusion matrix = [[226. 23. 14. 22.]

[60. 209. 7. 23.]

[64. 51. 153. 29.]

[51. 65. 27. 176.]]

INFO:KNN:On Validation Data

INFO:KNN:K = 5:

Accuracy = 0.500

Recall_per class = [0.5454545454545454, 0.65, 0.2972972972972973, 0.4827586206896552]

Precision_per class = [0.4897959183673469, 0.5416666666666666, 0.55, 0.42424242424242425]

macro_F1 = 0.486

confusion matrix = [[24. 6. 5. 9.]

[11. 26. 0. 3.]

[6. 13. 11. 7.]

[8. 3. 4. 14.]]

برای حالتی که پیش پردازش متنی برای حذف stopword ها داشته باشیم و فاصله گذاری بر حسب فاصله اقلیدسی بین بردارها باشد خروجی مطابق زیر است.

Output

ffBuilding tf idf table...

tf idf table built!

INFO:KNN:On Validation Data

INFO:KNN:K = 1:

Accuracy = 0.467

Recall_per class = [0.45454545454545453, 0.2, 0.40540540540540543, 0.9310344827586207]

Precision_per class = [0.7407407407407407, 0.8, 0.8333333333333334,

0.28421052631578947]

macro_F1 = 0.466

confusion matrix = [[20. 2. 2. 20.]

[4. 8. 0. 28.]

[2. 0. 15. 20.]

[1. 0. 1. 27.]]

INFO:KNN:On Validation Data

INFO:KNN:K = 3:

Accuracy = 0.473

Recall_per class = [0.7272727272727273, 0.1, 0.35135135135135137, 0.7586206896551724]

Precision_per class = [0.43243243243243246, 1.0, 0.9285714285714286, 0.3793103448275862]

macro_F1 = 0.435

confusion matrix = [[32. 0. 0. 12.]

[30. 4. 0. 6.]

[6. 0. 13. 18.]

[6. 0. 1. 22.]]

INFO:KNN:On Validation Data

INFO:KNN:K = 5:

Accuracy = 0.520

Recall_per class = [0.75, 0.3, 0.35135135135135137, 0.6896551724137931]

Precision_per class = [0.515625, 0.75, 0.8666666666666667, 0.36363636363636365]

macro_F1 = 0.504

confusion matrix = [[33. 0. 0. 11.]

[20. 12. 0. 8.]

[7. 1. 13. 16.]

[4. 3. 2. 20.]]

INFO:KNN:Best value for k: 5

INFO:KNN:On Train Data

INFO:KNN:K = 5:

Accuracy = 0.755

Recall_per class = [0.8666666666666667, 0.6889632107023411, 0.5824915824915825, 0.877742946708464]

Precision_per class = [0.6569148936170213, 0.8879310344827587, 0.8963730569948186, 0.7017543859649122]

macro_F1 = 0.752

confusion matrix = [[247. 6. 4. 28.]

[60. 206. 2. 31.]

[54. 10. 173. 60.]

[15. 10. 14. 280.]]

INFO:KNN:On Validation Data

INFO:KNN:K = 5:

Accuracy = 0.520 چ

Recall_per class = [0.75, 0.3, 0.35135135135135137, 0.6896551724137931]

Precision_per class = [0.515625, 0.75, 0.8666666666666667, 0.36363636363636365]

macro_F1 = 0.504

confusion matrix = [[33. 0. 0. 11.]

[20. 12. 0. 8.]

[7. 1. 13. 16.]

[4. 3. 2. 20.]]

به نظر می‌آید اضافه کردن پیش پرداز برای حذف stop words ها به دقت مدل کمک کرده است.

1.2 دسته بند Naive bayes

برای پیاده‌سازی این بخش کلاس NaiveBayesClassifierرا تعریف کردم. کد مربوط به این بخش در ضمیمه این سند در phase2/naive_bayes.py آمده است.

برای حالتی که پیش پردازش متنی نداشته باشیم خروجی مطابق زیر است.

Output

Building tf idf table...

tf idf table built!

INFO:Naive Bayes: On Validation Data

INFO:Naive Bayes:alpha = 1

Accuracy = 0.884

Recall_per class = [0.8786666666666667, 0.9693333333333334, 0.8293333333333334, 0.8573333333333333]

Precision_per class = [0.8978201634877384, 0.9284802043422733, 0.8650904033379694, 0.8416230366492147]

macro_F1 = 0.883

confusion matrix = [[659. 32. 30. 29.]

[13. 727. 3. 7.]

[30. 13. 622. 85.]

[32. 11. 64. 643.]]

INFO:Naive Bayes: On Validation Data

INFO:Naive Bayes:alpha = 2

Accuracy = 0.882

Recall_per class = [0.8786666666666667, 0.972, 0.824, 0.852]

Precision_per class = [0.8990450204638472, 0.9204545454545454, 0.8643356643356643, 0.8407894736842105]

macro_F1 = 0.881

confusion matrix = [[659. 34. 30. 27.]

[12. 729. 2. 7.]

[28. 17. 618. 87.]

[34. 12. 65. 639.]]

INFO:Naive Bayes: On Validation Data

INFO:Naive Bayes:alpha = 3

Accuracy = 0.879

Recall_per class = [0.876, 0.972, 0.8253333333333334, 0.844]

Precision_per class = [0.8963165075034106, 0.9204545454545454, 0.8573407202216067, 0.8406374501992032]

macro_F1 = 0.879

confusion matrix = [[657. 33. 32. 28.]

[11. 729. 3. 7.]

[28. 18. 619. 85.]

[37. 12. 68. 633.]]

INFO:Naive Bayes:Best value for alpha: 1

INFO:Naive Bayes: On Train Data

INFO:Naive Bayes:alpha = 1

Accuracy = 0.918

Recall_per class = [0.9101666666666667, 0.9835, 0.881, 0.8991666666666667]

Precision_per class = [0.9288994726994387, 0.9491716261862635, 0.8947190250507786, 0.8997665110073382]

macro_F1 = 0.918

confusion matrix = [[5461. 199. 227. 113.]

[53. 5901.   23.   23.]

[189.   60. 5286. 465.]

[176.   57. 372. 5395.]]

INFO:Naive Bayes: On Validation Data

INFO:Naive Bayes:alpha = 1

Accuracy = 0.884

Recall_per class = [0.8786666666666667, 0.9693333333333334, 0.8293333333333334, 0.8573333333333333]

Precision_per class = [0.8978201634877384, 0.9284802043422733, 0.8650904033379694, 0.8416230366492147]

macro_F1 = 0.883

confusion matrix = [[659. 32. 30. 29.]

[13. 727.   3.   7.]

[30. 13. 622. 85.]

[32. 11. 64. 643.]]

برای حالتی که پیش پردازش متنی برای stemming داشته باشیم خروجی مطابق زیر است.

Output

Building tf idf table...

tf idf table built!

INFO:Naive Bayes: On Validation Data

INFO:Naive Bayes:alpha = 1

Accuracy = 0.885

Recall_per class = [0.8853333333333333, 0.972, 0.8226666666666667, 0.8586666666666667]

Precision_per class = [0.9058663028649386, 0.9239543726235742, 0.8690140845070422, 0.8385416666666666]

macro_F1 = 0.884

confusion matrix = [[664. 31. 29. 26.]

[9. 729. 3. 9.]

[28. 16. 617. 89.]

[32. 13. 61. 644.]]

INFO:Naive Bayes: On Validation Data

INFO:Naive Bayes:alpha = 2

Accuracy = 0.880

Recall_per class = [0.8826666666666667, 0.9706666666666667, 0.816, 0.852]

Precision_per class = [0.8994565217391305, 0.9215189873417722, 0.864406779661017, 0.8342036553524804]

macro_F1 = 0.880

confusion matrix = [[662. 32. 30. 26.]

[10. 728. 2. 10.]

[31. 16. 612. 91.]

[33. 14. 64. 639.]]

INFO:Naive Bayes: On Validation Data

INFO:Naive Bayes:alpha = 3

Accuracy = 0.877

Recall_per class = [0.8826666666666667, 0.9666666666666667, 0.812, 0.848]

Precision_per class = [0.8994565217391305, 0.9154040404040404, 0.8613861386138614, 0.8313725490196079]

macro_F1 = 0.877

confusion matrix = [[662. 34. 29. 25.]

[11. 725. 4. 10.]

[30. 17. 609. 94.]

[33. 16. 65. 636.]]

INFO:Naive Bayes:Best value for alpha: 1

INFO:Naive Bayes: On Train Data

INFO:Naive Bayes:alpha = 1

Accuracy = 0.911

Recall_per class = [0.9026666666666666, 0.9818333333333333, 0.8673333333333333, 0.8933333333333333]

Precision_per class = [0.9201495073054706, 0.9463453815261044, 0.8888129803586678, 0.8882996353994034]

macro_F1 = 0.911

confusion matrix = [[5416. 205. 240. 139.]

[52. 5891.   24.   33.]

[226.   68. 5204. 502.]

[192.   61. 387. 5360.]]

INFO:Naive Bayes: On Validation Data

INFO:Naive Bayes:alpha = 1

Accuracy = 0.885

Recall_per class = [0.8853333333333333, 0.972, 0.8226666666666667, 0.8586666666666667]

Precision_per class = [0.9058663028649386, 0.9239543726235742, 0.8690140845070422, 0.8385416666666666]

macro_F1 = 0.884

confusion matrix = [[664. 31. 29. 26.]

[9. 729.   3.   9.]

[28. 16. 617. 89.]

[32. 13. 61. 644.]]

برای حالتی که پیش پردازش متنی برای حذف stop words داشته باشیم خروجی مطابق زیر است.

Output

ffBuilding tf idf table...

tf idf table built!

INFO:Naive Bayes: On Validation Data

INFO:Naive Bayes:alpha = 1

Accuracy = 0.887

Recall_per class = [0.9318181818181818, 1.0, 0.7297297297297297, 0.8620689655172413]

Precision_per class = [0.9111111111111111, 0.9302325581395349, 0.8709677419354839, 0.8064516129032258]

macro_F1 = 0.878

confusion matrix = [[41. 1. 2. 0.]

[0. 40. 0. 0.]

[2. 2. 27. 6.]

[2. 0. 2. 25.]]

INFO:Naive Bayes: On Validation Data

INFO:Naive Bayes:alpha = 2

Accuracy = 0.900

Recall_per class = [0.9318181818181818, 1.0, 0.7837837837837838, 0.8620689655172413]

Precision_per class = [0.9534883720930233, 0.9523809523809523, 0.8529411764705882, 0.8064516129032258]

macro_F1 = 0.892

confusion matrix = [[41. 1. 2. 0.]

[0. 40. 0. 0.]

[1. 1. 29. 6.]

[1. 0. 3. 25.]]

INFO:Naive Bayes: On Validation Data

INFO:Naive Bayes:alpha = 3

Accuracy = 0.893

Recall_per class = [0.9318181818181818, 0.975, 0.7837837837837838, 0.8620689655172413]

Precision_per class = [0.9534883720930233, 0.9512195121951219,

0.8285714285714286, 0.8064516129032258]

macro_F1 = 0.886

confusion matrix = [[41. 1. 2. 0.]

[0. 39. 1. 0.]

[1. 1. 29. 6.]

[1. 0. 3. 25.]]

INFO:Naive Bayes:Best value for alpha: 2

INFO:Naive Bayes: On Train Data

INFO:Naive Bayes:alpha = 2

Accuracy = 0.977

Recall_per class = [0.9649122807017544, 0.9933110367892977, 0.9764309764309764, 0.9717868338557993]

Precision_per class = [0.9892086330935251, 0.9867109634551495, 0.9508196721311475, 0.9810126582278481]

macro_F1 = 0.977

confusion matrix = [[275. 2. 7. 1.]

[0. 297. 2. 0.]

[1. 1. 290. 5.]

[2. 1. 6. 310.]]

INFO:Naive Bayes: On Validation Data

INFO:Naive Bayes:alpha = 2

Accuracy = 0.900

Recall_per class = [0.9318181818181818, 1.0, 0.7837837837837838, 0.8620689655172413]

Precision_per class = [0.9534883720930233, 0.9523809523809523, 0.8529411764705882, 0.8064516129032258]

macro_F1 = 0.892

confusion matrix = [[41. 1. 2. 0.]

[0. 40. 0. 0.]

[1. 1. 29. 6.]

[1. 0. 3. 25.]]

به نظر می‌رسد انجام پیش پردازش متنی از نظر stemming, lemmatization, stop word removal سبب بهبود دقت می‌شود.

1.3 دسته‌بند SVM

برای پیاده‌سازی این بخش کلاس SVM Classifierرا تعریف کردم و برای پیاده سازی مدل و فیت شدن آن به داده‌ها از کتابخانه sklearn استفاده کردم و سه پارامتر ۰.۵و ۱و ۱.۵را به عنوان C در Linear SVM در نظر گرفتم. کد مربوط به این بخش در phase2/svm.py آمده است.

خروجی این بخش مطابق زیر است.

Output

Building tf idf table...

tf idf table built!

INFO:SVM: On Validation Data

INFO:SVM:C = 0.5:

Accuracy = 0.213

Recall_per class = [0.0, 0.075, 0.0, 1.0]

Precision_per class = [0, 0.6, 0, 0.2]

macro_F1 = 0.11666666666666667

confusion matrix = [[0. 0. 0. 44.]

[0. 3. 0. 37.]

[0. 2. 0. 35.]

[0. 0. 0. 29.]]

INFO:SVM: On Validation Data

INFO:SVM:C = 1.0:

Accuracy = 0.267

Recall_per class = [0.022727272727272728, 0.175, 0.10810810810810811, 0.9655172413793104]

Precision_per class = [1.0, 0.4117647058823529, 0.6666666666666666, 0.2222222222222222]

macro_F1 = 0.20934882843517896

confusion matrix = [[1. 7. 2. 34.]

[0. 7. 0. 33.]

[0. 2. 4. 31.]

[0. 1. 0. 28.]]

INFO:SVM: On Validation Data

INFO:SVM:C = 1.5:

Accuracy = 0.360

Recall_per class = [0.13636363636363635, 0.35, 0.2702702702702703, 0.8275862068965517]

Precision_per class = [0.6666666666666666, 0.5185185185185185, 0.5, 0.2553191489361702]

macro_F1 = 0.3463616593805743

confusion matrix = [[6. 8. 6. 24.]

[2. 14. 1. 23.]

[1. 3. 10. 23.]

[0. 2. 3. 24.]]

INFO:SVM: On Train Data

INFO:SVM:C = 1.5:

Accuracy = 0.420

Recall_per class = [0.11228070175438597, 0.43478260869565216, 0.2727272727272727, 0.8181818181818182]

Precision_per class = [0.8, 0.5416666666666666, 0.675, 0.32625]

macro_F1 = 0.38356874732551516

confusion matrix = [[32. 37. 22. 194.]

[2. 130.   7. 160.]

[4. 27. 81. 185.]

[2. 46. 10. 261.]]

INFO:SVM: On Validation Data

INFO:SVM:C = 1.5:

Accuracy = 0.360

Recall_per class = [0.13636363636363635, 0.35, 0.2702702702702703, 0.8275862068965517]

Precision_per class = [0.6666666666666666, 0.5185185185185185, 0.5, 0.2553191489361702]

macro_F1 = 0.3463616593805743

confusion matrix = [[6. 8. 6. 24.]

[2. 14. 1. 23.]

[1. 3. 10. 23.]

[0. 2. 3. 24.]]

1.4 دسته‌بند Random Forest

برای پیاده‌سازی این بخش کلاس Random Forest Classifierرا تعریف کردم و برای پیاده سازی مدل و فیت شدن آن به داده‌ها از کتابخانه sklearn استفاده کردم و پارامترهای (۱۰و ۴۰) و (۱۵و ۶۰)را به ترتیب برای max_features و max_depths در نظر گرفتم. کد مربوط به این بخش در phase2/random_forest.py آمده است.

خروجی این بخش مطابق زیر است.

Output

Building tf idf table...

tf idf table built!

INFO:Random Forest: On Test Data

INFO:Random Forest: Accuracy = 0.820

Recall_per class = [0.7727272727272727, 0.925, 0.6756756756756757, 0.9310344827586207]

Precision_per class = [0.918918918918919, 0.9024390243902439, 0.8928571428571429, 0.6136363636363636]

macro_F1 = 0.816

confusion matrix = [[34. 3. 2. 5.]

[0. 37. 0. 3.]

[3. 0. 25. 9.]

[0. 1. 1. 27.]]

INFO:Random Forest: On Test Data

INFO:Random Forest: Accuracy = 0.833

Recall_per class = [0.8636363636363636, 0.975, 0.5945945945946, 0.896551724137931]

Precision_per class = [0.9743589743589743, 0.8666666666666667, 0.88, 0.6341463414634146]

macro_F1 = 0.821

confusion matrix = [[38. 2. 2. 2.]

[0. 39. 0. 1.]

[1. 2. 22. 12.]

[0. 2. 1. 26.]]

INFO:Random Forest:Best value for max depth: 60 , max feature: 15

INFO:Random Forest: On Train Data

INFO:Random Forest: Accuracy = 1.000

Recall_per class = [1.0, 1.0, 1.0, 1.0]

Precision_per class = [1.0, 1.0, 1.0, 1.0]

macro_F1 = 1.000

confusion matrix = [[285. 0. 0. 0.]

[0. 299. 0. 0.]

[0. 0. 297. 0.]

[0. 0. 0. 319.]]

INFO:Random Forest: On Validation Data

INFO:Random Forest: Accuracy = 0.833

Recall_per class = [0.8636363636363636, 0.975, 0.5945945945946, 0.896551724137931]

Precision_per class = [0.9743589743589743, 0.8666666666666667, 0.88, 0.6341463414634146]

macro_F1 = 0.821

confusion matrix = [[38. 2. 2. 2.]

[0. 39. 0. 1.]

[1. 2. 22. 12.]

[0. 2. 1. 26.]]

(2) خوشه‌بندی‌ها

2.1 خوشه‌بند K means

برای پیاده سازی این بخش کلاس K means را پیاده‌سازی کردم و برای به تصویر کشیدن خوشه‌ها از روش TSNE استفاده کرده و ابعاد کلمات به کارگرفته شده در سندرا به دو بعد کاهش دادم. کد مربوط به این بخش در ضمیمه این سند در phase2/kmeans.py آمده است.

خروجی به شرح تصویر زیر است.

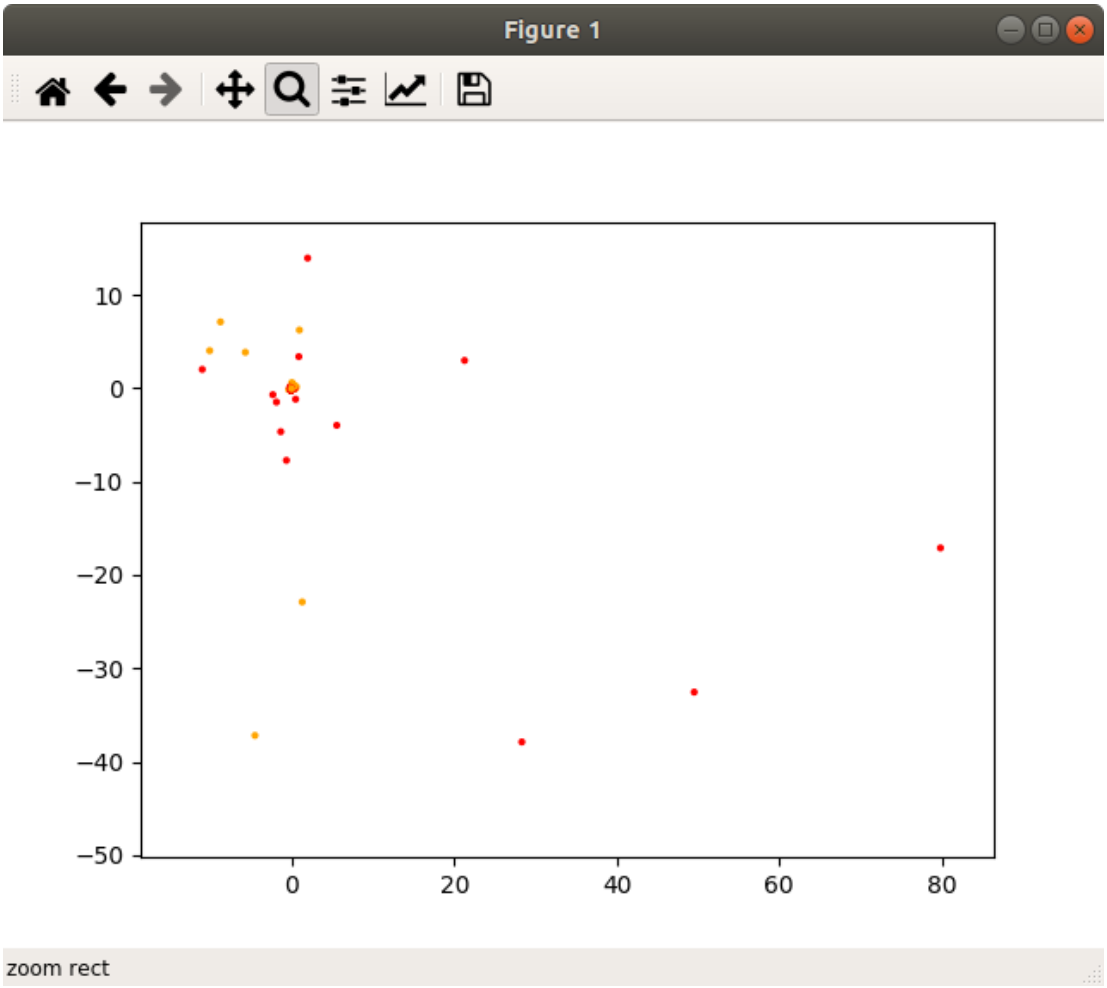

Illustration 1: K means classification