«به نام خدا»

تکلیف پنجم – سوال دوم – مرضیه علیدادی – 9631983

(کد های مربوط، در دو فرمت py. و ipynb. ضمیمه شده اند.)

.2

b) در این دیتاســـت missing value ای وجود ندارد. مقادیر Null ای وجود ندارد. همه ی attribute های input از نوع float هستند و نمی توانند دارای مقادیری از نوع string , ... (مثلا 'missed') به مفهوم missing value باشند.

داده های عددی را به این صورت normalize کردم:

	0	1	2	3	4
0	0.369056	0.883110	-0.286075	-0.045550	0
1	0.465045	0.835525	-0.251515	-0.149573	0
2	0.763780	-0.521232	0.380152	0.021031	0
3	0.301268	0.829982	-0.349606	-0.313278	0
4	0.050902	-0.688796	0.706823	-0.152873	0
...	...	...	...	...	...
1367	0.193595	0.643125	-0.691221	-0.266693	1
1368	-0.168665	-0.592375	0.786715	0.041512	1
1369	-0.165672	-0.594541	0.777189	-0.122680	1
1370	-0.230903	-0.543141	0.802981	-0.083084	1
1371	-0.645067	-0.166993	0.681179	0.303310	1

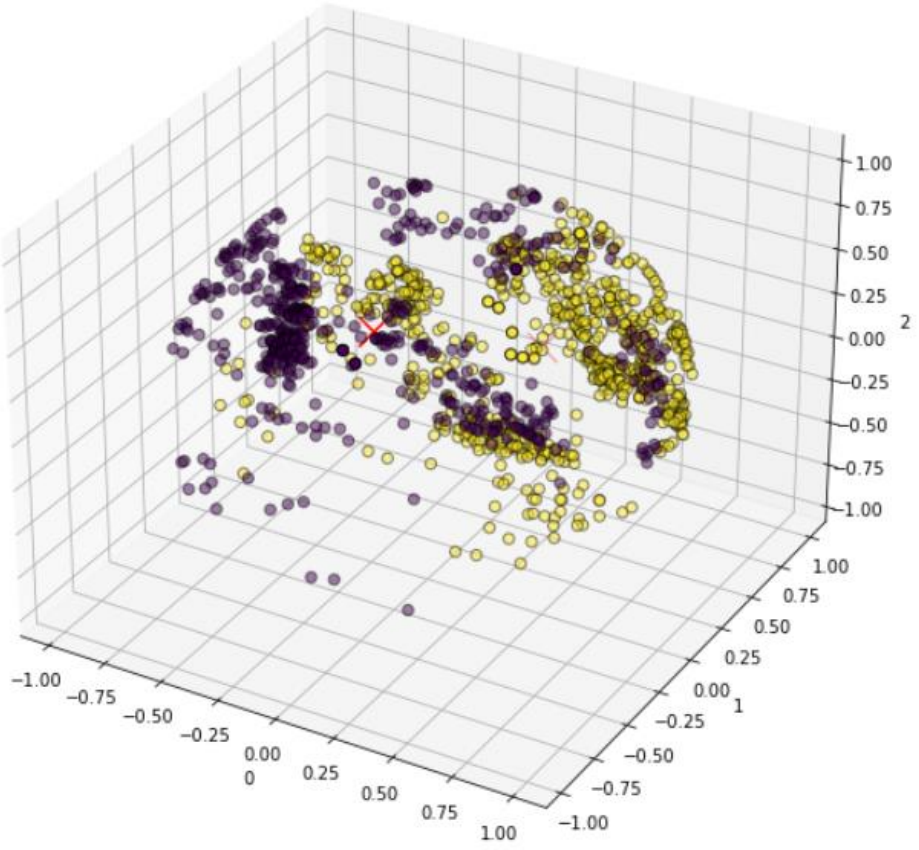
1372 rows × 5 columns

برای encode کردن متغیر Class، با توجه به اینکه جز input ها نیست و با توجه به اینکه در تحلیل ها اثر ندارد، حساسیتی روی مقادیری که می گیرد، نداریم. پس از Ordinal encoding ســـاده می توانیم اســـتفاده کنیم. ولی اینجا خودش مقادیرش به صورت عددی است؛ و نیازی به انکد کردن نداریم کلا.

(d

	0	1	2	3
0	0.015567	-0.412002	0.613780	0.063957
1	0.099362	0.711222	-0.172649	-0.228759

e)



f) پارامتر algorithm نوع الگوریتم مورد استفاده در kmeans را مشخص می کند.

در سه حالت قابل تنظیم است: auto و full و elkan. به صورت دیفالت روی auto تنظیم است.

الگوریتم EM-style کلاسیک، full است. الگوریتم elkan برای داده هایی که به طور مشخص دسته بندی می شوند، با استفاده از نابرابری مثلث، بسیار کارامد عمل می کند. با این حال ، به دلیل تخصیص یک آرایه اضافه تر، حافظه بیشتری نیاز است.

در حال حاضر، auto که دیفالت است، elkan را استفاده می کند. ممکن است در آینده تغییر کند.

g) الگوریتم k-means هدفش به دست آوردن centroid ها به گونه ای است، که اینرسی یا معیار جمع مربعات درون خوشه را به حداقل برساند.

```
kmeans.inertia_
```

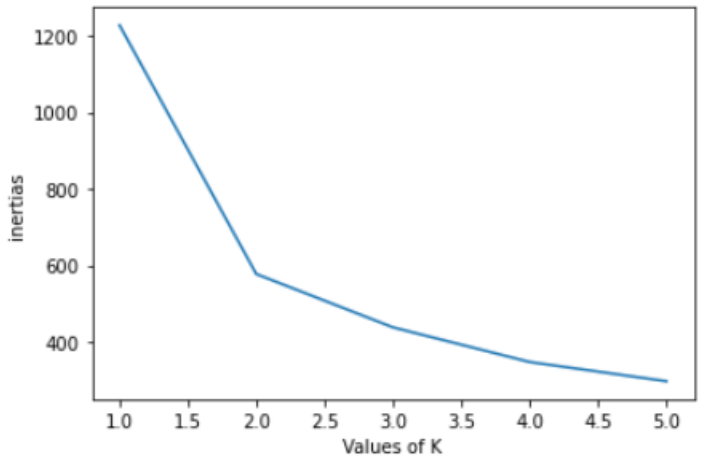
```
577.5323317567828
```

h)

```
[1226.8216989520497,
 577.5323317567828,
 438.3550696940430 4,
 347.961747756162,
 297.85265245237053]
```

همانطور که مشخص است، وقتی k برابر 2 است، همان نتیجه ی یکسان با بالا که k برابر 2 بود را دریافت کردیم.

i) نمودار:



بیشترین اختلاف در فاصله ی 1 و 2 اتفاق افتاده است.

برای تعیین تعداد بهینه خوشه ها، باید مقدار k را در نقطه ای که بعد از آن اینرسی به صورت خطی شروع به کاهش می کند، انتخاب کنیم. بنابراین برای داده های داده شده، نتیجه می گیریم که تعداد بهینه خوشه ها برابر **2** است.

j) برای استفاده از این شاخص، با توجه به اینکه 2 تا لیبل برای متغیر هدف وجود دارد،

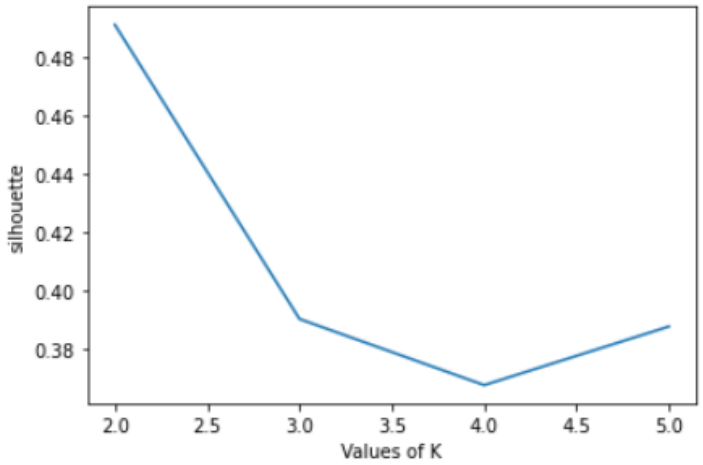
```
np.unique(kmeans.labels_)
```

```
array([0, 1])
```

نمی توان از k برابر با 1 استفاده کرد. پس، از 2 تا 5 را حساب می کنم:

```
[0.4914080638762628,
 0.3902349005548654,
 0.367506051883332,
 0.3856757344016 3155]
```

نمودار:



هر چقدر مقدار silhouette بیشـــتر باشـــد، یعنی مقدار k بهینه تر اســـت. بیشـــترین مقدار silhouette، مقدار بهینه ی سراسری را نشان می دهد. بنابراین برای داده های داده شده، نتیجه می گیریم که تعداد بهینه خوشه ها برابر **2** است.

k) با هر دو روش به این نتیجه ی یکسـان رسـیدیم، که تعداد خوشـه ها باید برابر با **2** باشـد. تا بهینه ترین نتیجه حاصـل شود.