

بسم الله الرحمن الرحيم

پروژه ۳ درس هوش مصنوعی  
دکتر فدایی و دکتر یعقوب زاده

مهندی وجہی

۸۱۰۱۰۱۵۵۸

# فهرست

3.....	مقدمه
4.....	خواندن داده ها
4.....	پیش پردازش و استخراج ویژگی ها
4.....	اهمیت استخراج ویژگی
4.....	استخراج ویژگی از تصاویر
5.....	پیش پردازش
5.....	پیاده سازی
5.....	خوش بندی
5.....	روش k mean
6.....	روش DBScan
7.....	مقایسه دو روش خوش بندی
7.....	تعیین k در k mean
8.....	کمیابه سازی k mean
9.....	پیاده سازی DBScan
10.....	مقایسه نتایج
10.....	کاهش بعد و PCA
11.....	ارزیابی نتایج
11.....	معیار silhouette
12.....	معیار homogeneity
13.....	روش های بهبود و توسعه
14.....	منابع

## مقدمه

هدف از این پروژه شناسایی گونه های مختلف گل ها از بین تصاویر مختلف آنها است. در این پروژه ما از یادگیری بدون نظارت استفاده می کنیم این روش در مواردی که حجم و تعداد داده ها زیاد است و داده های ما برچسب ندارند. این روش ها کاربرد های گوناگون دارد از جمله خوشه بندی داده ها که در این پروژه مورد استفاده قرار می گیرد. همچنین در این پروژه برای پردازش تصاویر از ویژگی VGG16 استفاده می شود. این ویژگی در واقع یک شبکه عصبی کانولوشنی ساده است که در ادامه با آن بیشتر آشنا می شویم.

# خواندن داده ها

در این قسمت ما لازم است مجموعه داده ی همراه پروژه را پردازش کنیم. این داده ها شامل ۲۱۰ تصویر مختلف از گل هاست. در آخر هم در یک فایل csv داده ها برای ارزیابی مدل برچسب خورده اند. در این قسمت تصاویر را خوانده و ابعاد ۲۲۴ در ۲۲۴ در می آوریم و سپس یک پیش پردازش اولیه روی تصاویر انجام می دهیم تا در مرحله بعد به VGG16 بدهیم. هم چنین برچسب های درست هم از داخل فایل csv می خوانیم و ذخیره می کنیم.

```
for filename in os.listdir(path):
    if filename.endswith('.png'):
        img_path = os.path.join(path, filename)
        img = load_img(img_path, target_size=(224,224))
        img = img_to_array(img).reshape(1,224,224,3)
        img = preprocess_input(img)
        images.append(img)
        names.append(filename)
```

## پیش پردازش و استخراج ویژگی ها

### اهمیت استخراج ویژگی

در پروژه های مختلف ما نیاز داریم موارد متفاوتی را بسنحیم هر کدام از این موارد از ویژگی ای استفاده می شود که بازنمایی مناسبی برای ما داشته باشد تا بتوانیم تحلیل مناسبی داشته باشیم اما اگر فقط بخواهیم روی پیکسل های بدون استخراج ویژگی کار کنیم بسیاری از موارد را نمی توان تحلیل کرد و درک مناسبی از تصویر به نمی دهد.

### استخراج ویژگی از تصاویر

از تکنیک های پردازش تصویر می توان به موارد زیر اشاره کرد:

- **تبديل فوريه:** در اين روش از تصویر تبديل فوريه گرفته می شود و آن را به اجزای سينوسی و كسينوسی تقسيم می کند. اين تبديل نوع گستته و پيوسته دارد ولی معمولا از تبديل فوريه گستته استفاده می شود. در اين تبديل عبارت سينوسی سه ضريب برای اندازه، فركانس و فاز دارد. اندازه نشان دهنده كنتراست، فاز نشان دهنده اطلاعات رنگی تصویر است و فركانس نشان دهنده روشناني تصویر است.
- **استخراج لبه تصویر:** روش های مختلفی برای اين کار وجود دارد. يکی از روش اين است که ما يك مربع ۳ در ۳ را از پیکسل ها را در نظر می گيریم و اگر در ردیف یا ستون ها اختلاف پیکسل بیش از يک مقدار خاصی بود آن را به عنوان لبه تشکیل می دهیم.

- تبدیل ۳ کانال رنگی به یک کانال: این ویژگی که میزان روشنی هر پیکسل را به ما می دهد در بعضی از پروژه ها برای کاهش میزان پردازش و عدم اهمیت رنگ ها استفاده می شود. روش استخراج آن آسان است نتها کافیست میانگین سه کانال رنگی را به دست آوریم و جایگزین کنیم.

## پیش پردازش

کار های مختلفی می توان برای پیش پردازش تصاویر انجام داد. از جمله آنها می توان به یکسان کردن ابعاد تصاویر اشاره کرد. این کار به ما کمک می کند که بتوانیم از ویژگی هایی که به ابعاد تصاویر وابسته هستند استفاده کنیم. یکی از کار های دیگری که می توان انجام داد حذف نویز از تصاویر است. این موضوع به ما کمک می کنیم که بتوانیم مدلی دقیق تر داشته باشیم همچنین می توان روی تصویر نرمال سازی انجام داد و ناهنجاری ها را از آن حذف کرد.

## پیاده سازی

با یک جست و جوی ساده در اینترنت کد لازم برای اجرای VGG16 با حذف لایه های تماماً متصل را به دست می آوریم.

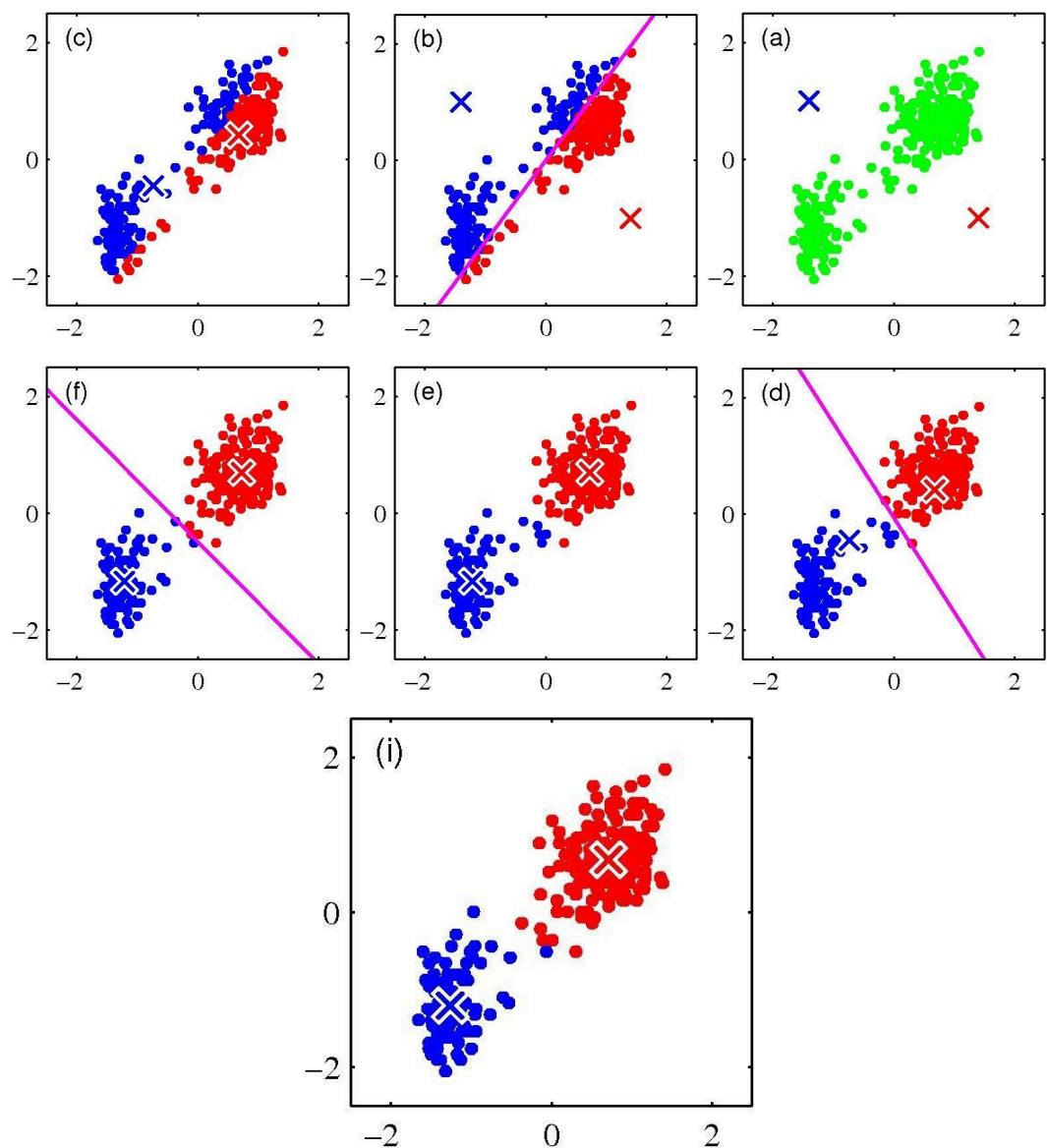
```
model = VGG16()
model = Model(inputs = model.inputs, outputs = model.layers[-2].output)

features = np.array([model.predict(i) for i in images]).reshape(-1,4096)
```

## خوش بندی

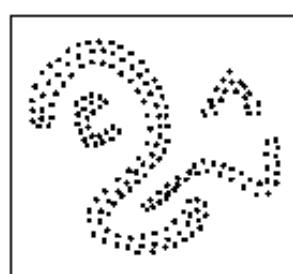
### k mean روش

ما در این روش سعی می کنیم برای داده های خود مرکز های خوش تعیین کنیم و داده ها را به این مرکز نسبت دهیم و با تکرار این کار خوش بندی و مرکز داده خود را بهتر کنیم. ابتدا به طور شناسی مرکز را در صفحه قرار می دهیم سپس داده ها را بر اساس فاصله نسبت می دهیم و در نهایت مرکز هر دسته را حساب می کنیم و مرکز را به نقاط جدید تغییر می دهیم. همین عمل را به طور تکرار شونده انجام می دهیم تا تغییر مرکز به صفر میل کند. تصاویر زیر نحوه اجرای الگوریتم را نشان می دهد.



## DBScan روش

بر خلاف روش قبلی ما در این روش داده های متراکم را در یک دسته قرار می دهیم حتی اگر این تراکم شکل کره ای نداشته باشد مثلا خوشه را ها در دسته زیر پیدا می کند.



برای این الگوریتم ۲ پارامتر تعریف می شود یکی فاصله را نشان می دهد یکی حداقل همسایه ها را. این الگوریتم بررسی می کند که یک نقطه در دایره ای با آن فاصله حداقل نقاط را دارد یا خیر و در صورتی که این شرط برقرار بود آنها را در یک دسته قرار می دهد.

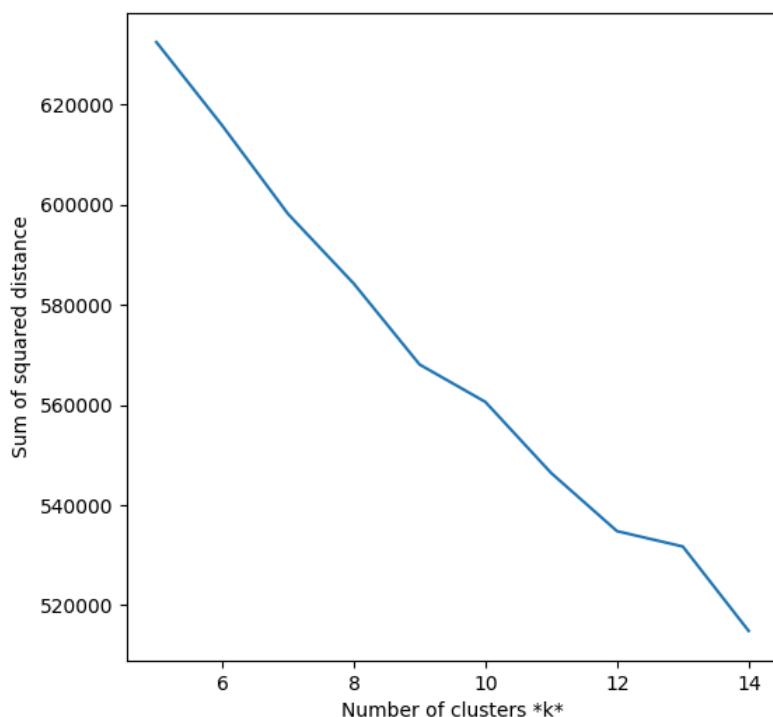
## مقایسه دو روش خوشه بندی

در جدول زیر مقایسه ای از این دو انجام دادیم.

DBScan	K mean	ویژگی مورد بررسی
متراکم (با هر شکلی)	کره ای با اندازه های مشابه	شكل مناسب داده ها
وابسته به داده ها	وابسته به پارامتر	تعداد خوشه ها
مناسب برای داده ها با حجم کم	مناسب برای داده ها با حجم زیاد	حجم داده ها
بی تاثیر	حساس	حساسیت به داده های پرت

## تعیین k در k mean

به صورت کلی در برای تعیین k می توان از نمودار Elbow استفاده کرد و نقطه شکست را به عنوان k مناسب انتخاب کرد در بعضی مواقع ما می دانیم که چند دسته داریم در اینجا نمودار Elbow را رسمی کنیم.



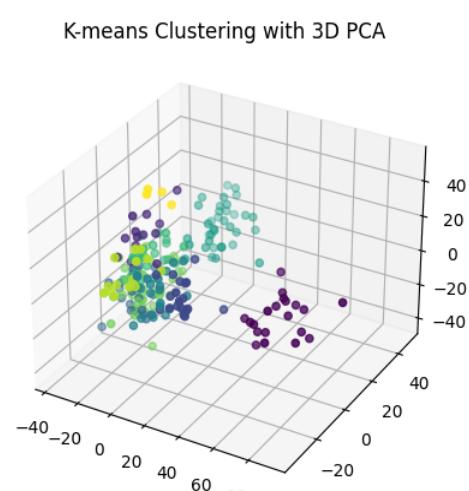
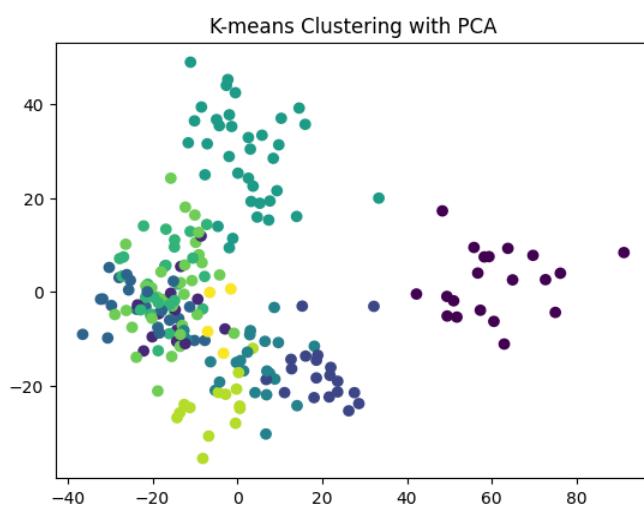
شاید با این تصویر بگوییم ۱۲ عدد مناسبی است اما با چند با رسم نمودار می بینیم که شکستگی خاص در نمودار نیست و نمی شود از آن استفاده کرد.

## k mean پیاده سازی

ما برای این پروژه k را برابر با تعداد دسته های فایل csv قرار دادیم تا صرفا عملکرد آن در تعیین خوشه ها بسنجدیم. پیچیدگی زیادی ندارد صرفا ازتابع مربوطه استفاده می کنیم.

```
kmeans = KMeans(n_clusters=len(df['label'].unique()))
kmeans.fit(features)
print(homogeneity_score(kmeans.labels_, real_labels))
print(silhouette_score(features, kmeans.labels_))
groups = convert_to_list_of_cluster(names, kmeans.labels_)
for i in groups.values():
    show_cluster(i)
```

در نهایت امتیاز تابع هموجنی ۶۶ درصد است و تابع سیلهوت ۰.۵ است. در تصویر زیر یک دسته ی خوب و بد را مشاهده می کنیم. درباره این معیار ها در ادامه صحبت می کنیم.



## پیاده سازی DBScan

به صورت زیر و امتحان چند باره پارامتر ها مقادیر آن را به دست می آوریم.

```
i,j = 54, 2
dbscan = DBSCAN(eps=i, min_samples=j)
dbscan.fit(features)
print(homogeneity_score(dbscan.labels_, real_labels))
groups = convert_to_list_of_cluster(names, dbscan.labels_)
for i in groups.values():
    show_cluster(i)
```

نتیجه مطلوب نیست و حجم زیادی از داده ها در یک دسته طبقه بندی می شوند.(داده نویز تشخیص داده می شوند)

بنابراین از pca استفاده می کنیم که کمی دقیق تر را کاهش دهیم تا بتوانیم داده های بیشتری را دسته بندی کنیم که برای پیدا کردن پارامتر مناسب می نویسیم.

```
for k in range(80,100):
    print(f'run for {k}')
    pca = PCA(n_components=k).fit_transform(features)
    for i in np.arange(45,60,0.1):
        for j in range(2,6):
            dbscan = DBSCAN(eps=i, min_samples=j)
            dbscan.fit(pca)
            groups = convert_to_list_of_cluster(names, dbscan.labels_)
            if 13 > len(groups.keys()) > 8 and len(groups[-1]) < 60:
                print(f'{i:.2f}, {j}, {k}, {len(groups[-1])},
{homogeneity_score(dbscan.labels_, real_labels):.2f}')
```

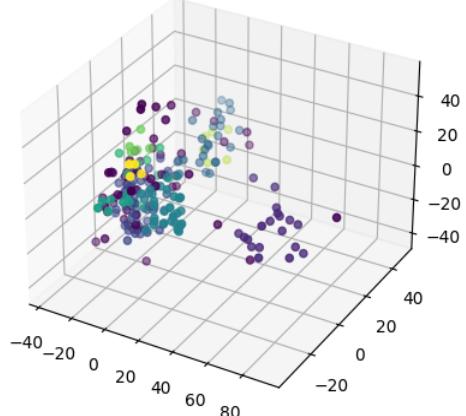
این کد را برای بازه های مختلف اجرا می کنیم. و در نهایت با پارامتر های مناسب کد را اجرا می کنیم.

```
i,j,k = 31.90, 4, 17
pca = PCA(n_components=k).fit_transform(features)
dbscan = DBSCAN(eps=i, min_samples=j)
dbscan.fit(pca)
print(homogeneity_score(dbscan.labels_, real_labels))
groups = convert_to_list_of_cluster(names, dbscan.labels_)
for i in groups.values():
    show_cluster(i)
```

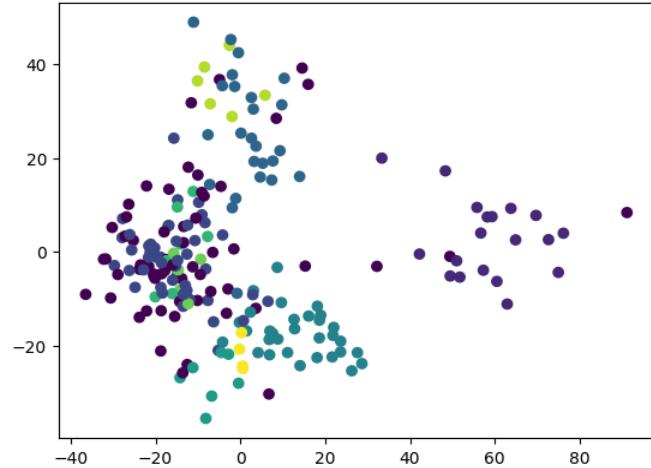
همogenتیک دقیق ۵۶ درصد می دهد و ۵۹ تصویر را نویز شناسایی می کند. تعدادی از دسته ها به صورت زیر است:



DBScan Clustering with 3D PCA



DBScan Clustering with PCA



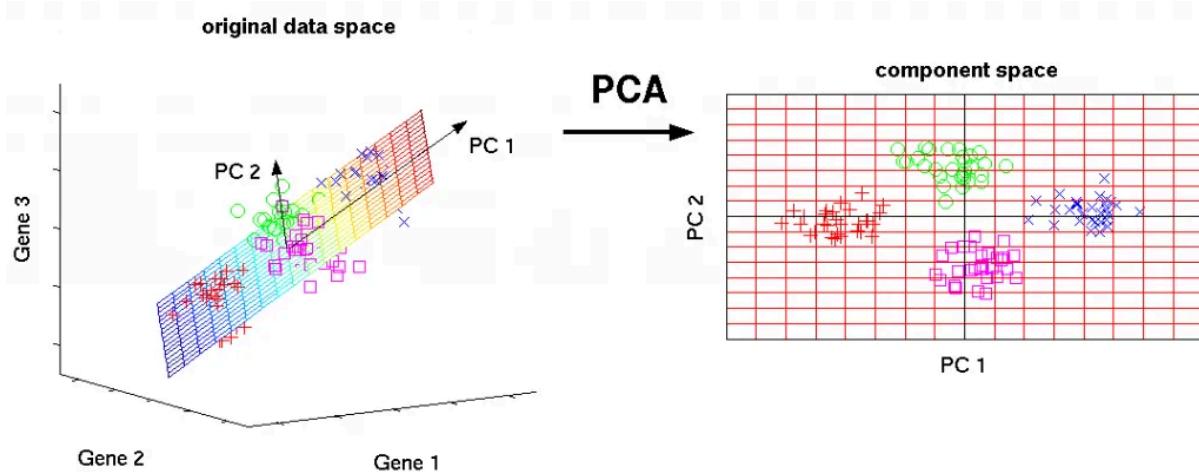
## مقایسه نتایج

در این پروژه به نظر می آید که kmean عملکرد بهتری دارد و با دقت ۶۵ درصدی همه داده ها را دسته بندی می کند اما در دسته های DBScan به نظر می آید اشتباه کمتر است و فقط موارد مطمئن رو قرار داده.

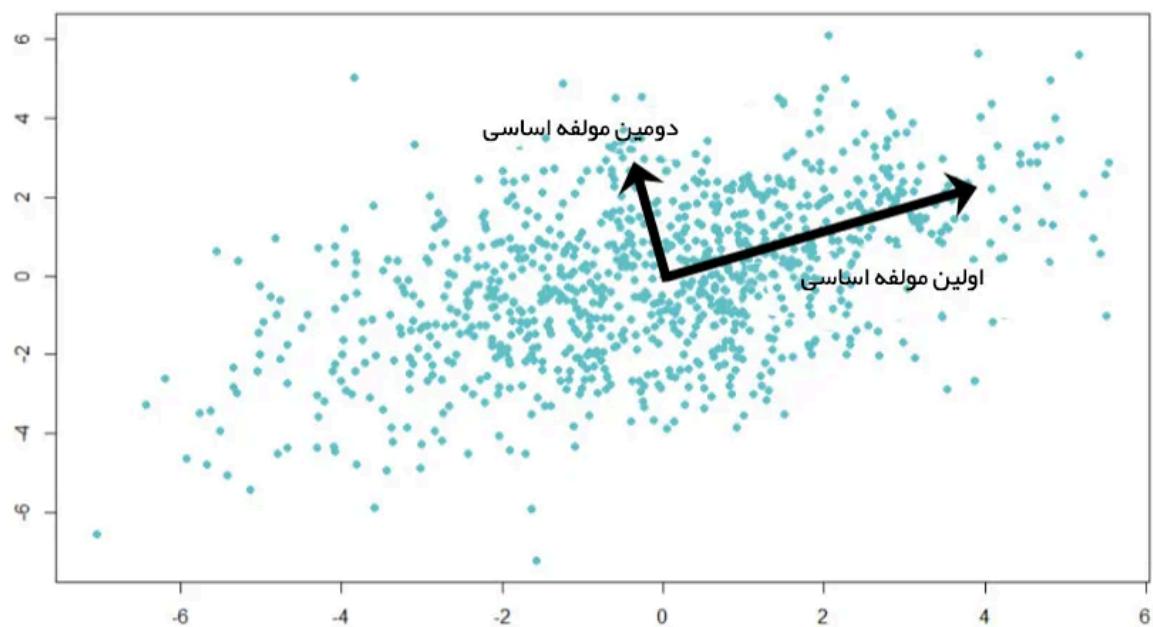
## کاهش بعد و PCA

در این روش از ابعاد ویژگی ما کم می شود برای این که بتوانیم مدل را نمایش دهیم یا پردازش ها را کمتر کنیم. برای این کار ما باید ابعاد کم ارزش را حذف کنیم. ابعاد کم ارزش ابعادی هستند که داده ی خاصی

ندازند و برای بیشتر داده ها تقریباً ثابت است و واریانس داده ها در آن کم است. تصویر زیر گویا عملکرد اینتابع است.



این کار با محاسبه واریانس ها انجام می شود و ابعادی که واریانس بیشتری دارند نگهداری می شود.



Pca برای الگوریتم های استفاده شده و نتایج رسم شده.

## ارزیابی نتایج

### معیار silhouette

این معیار میزان پیوستگی درون خوشه ای و تفکیک پذیری خوشه ها را نشان می دهد. برای محاسبه آن ابتدا فاصله هر نقطه را از باقی نقاط خوشه خود حساب می کنیم.

$$a(i) = \frac{1}{n_i} \sum_{l=1}^{n_i} d(x_i, x_l)$$

حال فاصله هر نقطه خوش با نزدیک ترین خوش به آن (از نظر میانگین فاصله) حساب می کنیم.

$$b(i) = \min_{1 \leq l \leq k} \frac{1}{n_l} \sum_{y_m \in C_l} (d(x_i, y_m))$$

در نهایت مقدار زیر را حساب می کنیم.

$$s(i) = \frac{b(i) - a(i)}{\max(b(i), a(i))}$$

این عدد مقداری بین ۰ و ۱ است و هر مقدار که به ۱ نزدیک باشد یعنی مدل بهتر است و خوشها مربوط تر هستند. در واقع هر مقدار که این معیار کم باشد نشان می دهد که نقاط درون خوش ارتباط کمی با خوش دارند و قابل تفکیک از باقی خوشها نیستند.

## معیار homogeneity

همانطور که از نام آن پیداست این معیار همگنی خوشها را نشان می دهد یعنی چقدر در یک خوش واقعاً یک دسته قرار دارد.  $a$  نشان دهنده داده متعلق به کلاس  $C$  و خوش  $k$  هست.

$$h = 1 - \frac{H(C, K)}{H(C)} H(C, K) = - \sum_{k=1}^K \sum_{c=1}^C \frac{a_{ck}}{N} \log\left(\frac{a_{ck}}{\sum_{c=1}^C a_{ck}}\right)$$

$$H(C) = - \sum_{c=1}^C \frac{\sum_{k=1}^K a_{ck}}{C} \log\left(\frac{\sum_{k=1}^K a_{ck}}{C}\right)$$

## روش های بهبود و توسعه

- پیش پردازش های بیشتر
- حذف نویز تصاویر
- استفاده از مدل های پیچیده تر و بهتر از VGG16
- استفاده از مدل های خوشه بندی دیگر و بررسی آنها
- طراحی و استخراج ویژگی های مناسب این مسئله خاص

## منابع

- [پردازش تصویر جست و جه کاربردی دارد؟](#)
-  lec12\_clustering.pptx
- [scikit-learn.org](#)
- [Difference between K-Means and DBScan Clustering - GeeksforGeeks](#)
- [Should I delete last 7 layers of VGG16 as I am going to use it as a pretrained model for a signature verification task? - Stack Overflow](#)
- [How to cluster images based on visual similarity | by Gabe Flomo | Towards Data Science](#)
- [راهنمای عملی به همراه کد نویسی در پایتون و - \(PCA\) تحلیل مولفه اساسی R](#)
- [\(Internal\) معیارهای درونی – \(Clustering Performance\) روش های ارزیابی نتایج خوشبندی فرادرسن - مجله – Index](#)
- [ML | V-Measure for Evaluating Clustering Performance - GeeksforGeeks](#)