به نام خدا

دانشگاه تهران

دانشکده مهندسی برق و کامپیوتر

یادگیری ماشین

گزارش پروژه نهایی

نام و نام خانوادگی	کامیار رحمانی نوید رزاقی محمد مشرقی
شماره دانشجویی	810199422 810199424 810199492
تاریخ ارسال گزارش	1402/04/12

فهرست گزارش سوالات

تمیز کردن داده و استخراج ویژگی 4

تمیز کردن داده 4

پیش پردازش 4

Resize 4

Label correcting 4

استخراج ویژگی 5

میانگین رنگ های RGB 5

Saturation 5

Hue 5

Watermark 5

Quantization table 6

ELA(error level analysis) 7

Fourier 7

Noise analysis 8

Wavelet 8

Luminance Gradient: 9

Local Binary Pattern: 12

HOG features: 16

Gray-Level Co-occurrence Matrix (GLCM): 17

سوال سوم: طبقه بندی 27

طبقه بندی با استفاده از داده های استخراج شده ی خودمان 27

طبقه بندی با استفاده از درخت تصمیم 27

طبقه بندی با استفاده از درخت تصمیم(max depth = 3) 29

طبقه بندی با استفاده از طبقه بند SVM 30

طبقه بندی با استفاده از logistic regression 33
طبقه بندی با استفاده از LDA 34
طبقه بندی با استفاده از کاهش بعد به روش PCA و سپس استفاده از طبقه بند SVM 34
طبقه بندی با استفاده از داده های استخراج شده توسط دستیار آموزشی 35
طبقه بندی با استفاده از درخت تصمیم 35
طبقه بندی با استفاده از LDA 38
کاهش بعد به روش PCA و سپس طبقه بندی به روش SVM 38
سوال چهارم: خوشه بندی 39
خوشه بندی با روش K-means 39
خوشه بندی با روش GMM 41
خوشه بندی با روش K-means(TA data) 44
خوشه بندی با روش GMM(TA) 48
خوشه بندی با روش K-means و بررسی چگونگی تفکیک کوه و دریا و جنگل: 52

تمیز کردن داده و استخراج ویژگی

تمیز کردن داده

پیش پردازش

Resize

در این پروژه برای اینکه بتوانیم سرعت عمل و دقت را بالا ببریم رزولیشن عکس ها را یکی و همه را 1024* 1024 پیکسل می کنیم در این مرحله دو عکس نتوانسته رد بشه و بعضی عکس حتی کدینگشون RGB نبود بعضی از آنها RGBA یا P که آنها را هم تبدیل به RGB کردیم و بعد با همان اسم در پوشه دیگر سیو کردیم.

و بعد از چک کردن آن دیدیم عکس ها خراب است. و بعد از آن با یک کد دیگر چک کردیم که تمام عکس همان سایز خواسته شده خودمان باشد.

```
Skipped invalid file: 810199515_real_none_jungle_1.jpeg - cannot identify
                                                                image file
Skipped invalid file: 810199515_real_none_jungle_10.jpeg - cannot identify
                                                                image file
```

Label correcting

در خوندن اسم های عکس برای لیبل زدن فیک یا ریل یا دسته بندی عناصر بعضی از کلمات را حروف اول را بزرگ یا کوچک نوشتند مثلا برای فیک ما دو کلمه داشتیم (Fake & fake) برای کلمه ریل داشتیم (Real & real) برای دسته بندی عکس ها هم برای کوهستان (Mountain, mountain) یا برای جنگل (forest , jungle , jungle ,Jungle) یا برای دریا (sea , Sea ,see) داشتیم و برای اینکه کجا درست شدن اسم های مختلفی داشتیم مثل (dalle,Dalle,dallE,dall_e) و غیره که این موارد را تصحیح کردیم.

روش درست کردن لیبل ها به این صورت است که اول لیبل ها را چه درست چه غلط ذخیره می کنیم و در همان کد پایتون if و for تمامی آنها را می گردیم و انواع لیبل ها را پیدا می کنیم و سپس وقتی دیدیم برا یک لیبل مخصوص چند حالت داریم یک حالت انتخاب می کنیم و بقیه آنها را به آن اسم ذخیره می کنیم.

در بعضی موارد دانشجویان به جای _ از – استفاده کردند و باید برا تشخیص آن را هم هندل کرد.

حال بعد از این وقتی ویژگی ها استخراج شدند و فایل حاوی این فیچر ها تهیه شد به سراغ بعضی داده ها که ممکنه آن فیچر را نداشته باشد یا جواب نامناسب به ما داده باشد می رویم و آن را حذف می کنیم تا در فرایند آموزش و تست مشکلی نداشته باشیم.

استخراج ویژگی

ویژگی هایی که خودمان از داده ها استخراج کردیم :

میانگین رنگ های RGB

در اینجا سه پارامتر به که از میانگین گرفتن رنگ های صفحه یعنی قرمز ، آبی و سبز بدست آمده است . این سه فیچر اگر به تنهایی آموزش دهیم اصلا و البدا دقت خوبی نخواهند داشت. اما وقتی با با فیچر های دیگه ای باشند می توانند دقت آموزش را بالا ببرند و مفید واقع شوند.

Saturation

اشباع به شدت یا خالصی رنگ ارتباط دارد. این مؤلفه میزان خاکستری رنگ موردنظر را تعیین می‌کند. یک رنگ با اشباع کامل، شاداب و روشن است، در حالی که رنگی با اشباع کمتر، ملایم‌تر و نزدیکتر به خاکستری به نظر می‌رسد. معمولاً اشباع به صورت درصدی اندازه‌گیری می‌شود، به طوری که ۰ درصد به یک رنگ کاملاً با اشباع کم (خاکستری) اشاره دارد و ۱۰۰ درصد به رنگی با بیشترین شدت و خالصی اشاره می‌کند.

Hue

این ویژگی به نسبت رنگ و تناسب و مقادیر رنگ های اصلی اشاره می کند به طوری که ۰ درجه رنگ قرمز را نمایان می‌کند، ۱۲۰ درجه رنگ سبز را نشان می‌دهد و ۲۴۰ درجه رنگ آبی است. با تغییر مقدار رنگ، می‌توانید از رنگ‌های مختلف طیف رنگی عبور کنید.

Watermark

در بعضی از عکس های فیک که توسط Dall.E درست شده در پایین سمت چپ و راست آن واتر مارک هست که میتوان با استخراج آن بخش هایی از عکس های فیک را بدست بیاوریم و دقت مدل را بالا ببریم برای این قسمت نواحی هر قسمت که رنگ و موقعیت آن ثابت هست را پیدا و با میانگین گرفتن و گذاشتن آن در یک بازه رنگی تشخیص می دهیم که آیا واتر مارک دارد یا نه.

Figure1 watermark 1

Figure2 watermark 2

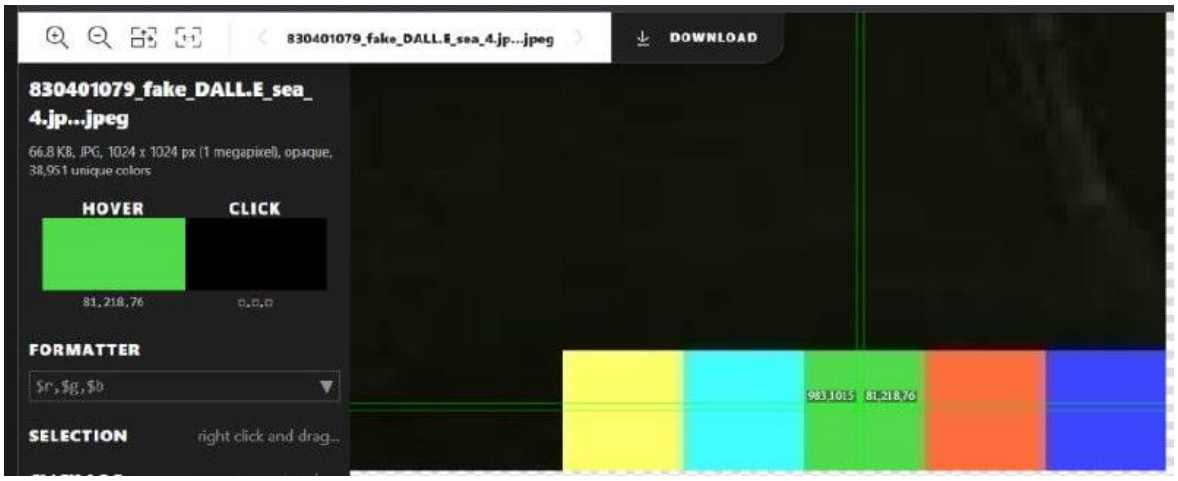


Figure3 finding watermark position

Quantization table

جداول کوانتیزاسیون (Quantization Tables) بخشی اساسی در الگوریتم‌های فشرده‌سازی تصاویر و ویدیوها مانند JPEG (گروه مشترک تخصصی عکاسی) هستند. این جداول برای کاهش حجم داده‌ها مورد استفاده قرار می‌گیرند و از محدودیت‌های حس بصری انسان بهره می‌برند.

در زمینه‌ی فشرده‌سازی تصاویر و ویدیوها، کوانتیزاسیون فرایندی است که یک محدوده‌ی مقادیر پیوسته را به یک مجموعه‌ای از مقادیر گسسته نگاشت می‌دهد. فرایند کوانتیزاسیون با کاهش دقت داده‌ها اطلاعاتی را از دست می‌دهد. با این حال، این افت اطلاعات معمولاً به طور غیر قابل تشخیص یا قابل قبول برای چشم انسان است.

در این پروژه ما با میانگین گرفتن و انحراف معیار گرفتن از ماتریس های تصاویر داده شده می توانیم بفهمیم که عکس مورد نظر فیک یا واقعی است معمولا عکس های فیک بیشترشان 1024 در 1024 و نحوه فشرده سازی شبیه هم هست که می توان از این قابلیت استفاده کرد و دقت خوبی بدست آورد.

نکته : در پیش پردازش عکس ها را همه یک سایز کردیم و در این روش باعث می شود مقادیر ماتریس یکسان شود پس این عملیات برا عکس هایی است که پیش پردازش روی آن ها صورت نگرفته است.

روش محاسبه ماتریس ها بر اساس الگوریتم های فشرده سازی شدش که در کتابخانه PIL روش آن هست.

ELA(error level analysis)

تجزیه و تحلیل سطح خطا (ELA) امکان شناسایی مناطقی را در یک تصویر که در سطوح مختلف فشرده سازی قرار دارند، می دهد. با تصاویرJPEG ، کل تصویر باید تقریباً در یک سطح باشد. اگر بخشی از تصویر در سطح خطای قابل توجهی متفاوت باشد، احتمالاً نشان دهنده یک تغییر دیجیتال است.

ELA تفاوت در نرخ فشرده سازی JPEG را برجسته می کند. مناطق با رنگ‌بندی یکنواخت، مانند آسمان آبی جامد یا دیوار سفید، احتمالاً نسبت به لبه‌های با کنتراست بالا، نتیجه ELA کمتری (رنگ تیره‌تر) خواهند داشت.

روش استخراج ویژگی :

- ✓ بارگذاری تصویر: از کتابخانه هایی مانند OpenCV یا PIL برای خواندن تصویر در کد خود می توانیم استفاده کنیم.
- ✓ انجام ELA: تکنیک ELA را روی تصویر اعمال می کنیم. ELA شامل ایجاد یک نسخه اصلاح شده از تصویر با ذخیره آن با کیفیتی خاص و سپس محاسبه تفاوت بین تصویر اصلاح شده و تصویر اصلی است. این فرآیند قسمت هایی از تصویر را که ممکن است تحت فشرده سازی یا دستکاری قرار گرفته اند برجسته کند.
- ✓ تبدیل تصاویر: هم تصویر اصلی و هم تصویر ELA را به یک فضای رنگی(معمولاً در مقیاس خاکستری) تبدیل می کنیم. این مرحله ثبات در تجزیه و تحلیل بعدی را تضمین می کند.
- ✓ محاسبه تفاوت پیکسل: تفاوت مطلق پیکسل بین تصویر اصلی و تصویر ELA را محاسبه می کنیم. این مرحله بر مناطقی از تصویر تأکید می‌کند که مقادیر پیکسل‌ها به‌طور قابل‌توجهی تغییر کرده‌اند.
- ✓ محاسبه ویژگی ها: ویژگی ها را از تصویر تفاوت پیکسل استخراج می کنیم. این ویژگی ها می توانند شامل معیارهای آماری مانند میانگین، انحراف استاندارد، حداکثر مقدار، حداقل مقدار یا ویژگی های پیچیده تر مانند توصیفگرهای بافت یا الگوهای باینری محلی (LBPs) باشند. این ویژگی‌ها ویژگی‌های تصویر ELA را نشان می‌دهند که مناطق بالقوه دستکاری را نشان می‌دهد.
- ✓ در نهایت، ویژگی‌های استخراج‌شده را به صورت لیست برای کارهای طبقه‌بندی return میکنیم.

Fourier

تحلیل فوریه می تواند به عنوان بخشی از فرآیند تشخیص تصاویر ساختگی استفاده شود. با تجزیه و تحلیل محتوای فرکانس یک تصویر با استفاده از تبدیل فوریه، اختلافات یا ناهنجاری ها در حوزه فرکانس می تواند دستکاری های بالقوه تصویر را نشان دهد.

روش استخراج ویژگی :

- ✓ بارگیری و پیش پردازش تصویر: با بارگیری تصویری که می خواهیم تجزیه و تحلیل کنیم با استفاده از کتابخانه ای مانند OpenCV یا PIL شروع می کنیم. اگر تصویر رنگی است، تصویر را به مقیاس خاکستری تبدیل می کنیم، زیرا تجزیه و تحلیل فوریه معمولاً روی تصاویر با مقیاس خاکستری انجام می شود.
- ✓ انجام تبدیل فوریه: تبدیل فوریه را روی تصویر خاکستری اعمال می کنیم. تبدیل فوریه تصویر را از حوزه فضایی به حوزه فرکانس تبدیل می کند. از الگوریتم تبدیل فوریه سریع (FFT) استفاده می کنیم که برای محاسبه تبدیل فوریه کارآمد است.
- ✓ Shift the Fourier Spectrum: پس از اعمال تبدیل فوریه، مولفه فرکانس صفر را به مرکز طیف منتقل می کنیم. این مرحله برای تجسم و تجزیه و تحلیل محتوای فرکانس تصویر به درستی ضروری است.
- ✓ محاسبه اندازه طیف: با گرفتن اندازه از ضرایب فوریه مختلط، اندازه طیف را محاسبه می کنیم. اندازه طیف نشان دهنده دامنه فرکانس های مختلف موجود در تصویر است.
- ✓ استخراج ویژگی ها: استخراج ویژگی ها از اندازه طیف. می توانیم معیارهای آماری مختلفی مانند میانگین، انحراف معیار، حداکثر مقدار یا مقادیر صدک را به عنوان ویژگی در نظر بگیریم. این ویژگی ها توزیع دامنه های فرکانس را در تصویر ثبت می کنند.
- ✓ در نهایت، ویژگی‌های استخراج‌شده را به صورت لیست برای کارهای طبقه‌بندی return میکنیم.

Noise analysis

تجزیه و تحلیل نویز می تواند یک تکنیک مفید برای تشخیص تصاویر ساختگی و واقعی باشد. با بررسی ویژگی های نویز یک تصویر، می توان ناهماهنگی ها یا ناهنجاری هایی را که ممکن است نشان دهنده دستکاری یا دستکاری تصویر باشد، شناسایی کرد. برای استخراج نویز یک تصویر ، از تابع $np.std(image)$ استفاده کردیم و انحراف معیار پیکسل های عکس را بدست آوردیم .

Wavelet

تجزیه و تحلیل wavelet می تواند به عنوان یک تکنیک موثر برای تشخیص تصاویر ساختگی استفاده شود. این روش شامل تجزیه یک تصویر به اجزای فرکانس مختلف با استفاده از تبدیل wavelet و تجزیه و تحلیل ضرایب حاصل است.

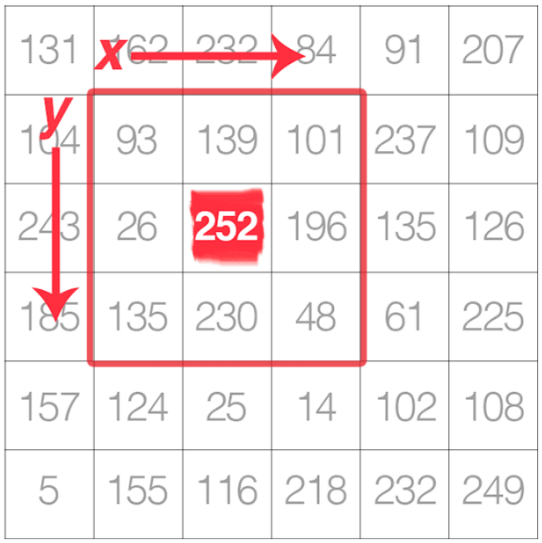
روش استخراج ویژگی :

- ✓ بارگیری تصویر

- ✓ Convert to Grayscale(تبدیل کردن عکس نمونه به عکس grayscale): اگر تصویر رنگی است، تصویر را به مقیاس خاکستری تبدیل می کنیم. تجزیه و تحلیل wavelet معمولاً بر روی تصاویر در مقیاس خاکستری انجام می شود.
- ✓ Apply Wavelet Transform: یک تبدیل wavelet را روی تصویر خاکستری اعمال می کنیم. تبدیل wavelet تصویر را به باندهای فرکانسی مختلف تجزیه می کند و اطلاعات مکانی و فرکانسی را ضبط می کند.
- ✓ انتخاب نوع wavelet و سطح تجزیه: یک نوع wavelet خاص (مانند Haar، Daubechies، Symlet) و سطح تجزیه دلخواه را بر اساس ویژگی های تصویر و سطح جزئیاتی که می خواهیم ثبت کنیم، انتخاب می کنیم.
- ✓ بدست آوردن ضرایب wavelet: ضرایب wavelet را در هر سطح تجزیه استخراج می کنیم. این ضرایب نشان دهنده ضرایب تقریبی و جزئیات تصویر در مقیاس ها و جهت گیری های مختلف است.
- ✓ محاسبه ویژگی‌های wavelet: از ضرایب wavelet، ویژگی های رایج عبارتند از میانگین، انحراف معیار، انرژی، آنتروپی، یا توصیف بافت را استخراج میکنیم.
- ✓ در نهایت، ویژگی‌های استخراج‌شده را به صورت لیست برای کارهای طبقه‌بندی return میکنیم.

Luminance Gradient:

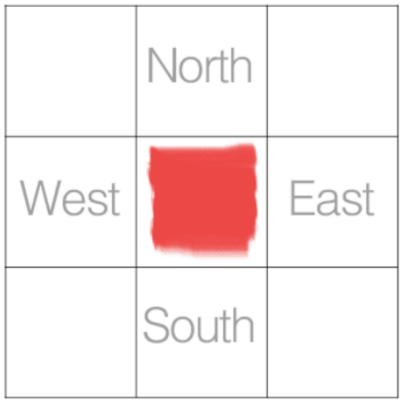
این ویژگی نشان دهنده تغییرات روشنایی(brightness) در یک تصویر می‌باشد. این ویژگی یک عامل مهم در تشخیص ناپیوستگی ها یا موارد غیرطبیعی موجود در تصویر است.



عکس هایی که توسط شبکه های GAN تولید شده است در این ویژگی تفاوت هایی با عکس های واقعی دارند. برای مثال یکی از اصلی ترین تفاوت ها این است که تصاویر تولید شده توسط شبکه های عمیق، اغلب الگوهای مصنوعی از خود نشان می‌دهند. برای مثال در تصاویر تولید شده توسط این شبکه‌ها تغییرات ناگهانی در روشنایی تصاویر دیده می‌شود یا الگوهای تکراری ای را از خود نشان می‌دهند که در تصاویر واقعی دیده نمی‌شود.

image gradient برای edge detection استفاده می‌شود. با محاسبه آن برای عکس های مختلف می‌توانیم به شکلی edge های موجود یا به عبارت دیگر تغییرات درخشندگی در تصاویر real و fake را بدست آوریم.

برای بدست آوردن پارامتری که نشان دهنده image gradient باشد ابتدا تصاویر را gray کردیم. برای محاسبه گرادیان باید تفاوت شدت(نور) هر پیکسل با پیکسل دیگری را در جهت مشخصی پیدا کنیم.



تصویر مقابل را در نظر بگیرید که هر پیکسل مربوط به تصویر gray چهار همسایگی در اطراف خود دارد.

فرض کنید در راستای محورهای x و y تغییرات شدت نور را بررسی می‌کنیم بنابراین intensity مربوط به هر کدام از پیکسل های اطراف پیکسل قرمز به صورت زیر بدست خواهد آمد:

- **East:** $I(x+1, y)$
- **North:** $I(x, y-1)$
- **West:** $I(x-1, y)$
- **South:** $I(x, y+1)$

بنابراین تغییرات در intensity در جهت محور x و y به شکل زیر بدست خواهد آمد:

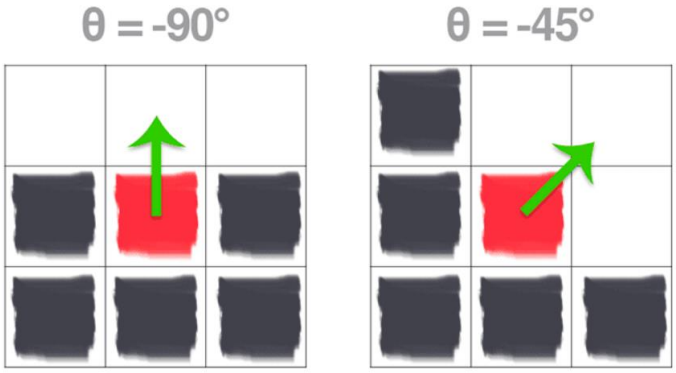
$$G_x = I(x+1, y) - I(x-1, y) \qquad G_y = I(x, y+1) - I(x, y-1)$$

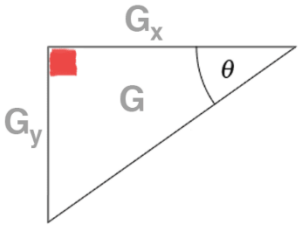
برای محاسبه gradient یک تصویر باید دو پارامتر gradient orientation و gradient magnitude را بدست آوریم.

gradient orientation: نشان می‌دهد که نشان دهنده مقدار تغییر شدت نور در تصویر است.

gradient magnitude: جهت تغییر شدت نور را نشان می‌دهد.

برای مثال در تصویر زیر جهت تغییر شدت نور در یک تصویر gray به کمک زاویه قابل بیان است:





برای محاسبه gradient magnitude کافی است که از تغییرات گرادیان در هر دو جهت x و y استفاده کنیم و سپس با ی محاسبه ساده ریاضی آن را بدست آوریم:

$$G = \sqrt{G_x^2 + G_y^2}$$

برای محاسبه gradient orientation از رابطه زیر استفاده می‌کنیم:

$$\theta = \arctan2(G_y, G_x) \times \left(\frac{180}{\pi}\right)$$

در رابطه فوق از atan2 استفاده می‌کنیم چون تابع $\tan^{-1}$ در بعضی مقادیر تعریف نشده است.

روش بدست آوردن گرادیان یک روش kernel-based است در واقع طبق شکل روبرو در هر کرنل دو پارامتر gradient magnitude و gradient orientation را بدست می‌آوریم.

در این پروژه از دو کرنل sobel و scharr استفاده کردیم:

کرنل sobel خودش از دو کرنل تشکیل شده است. یکی در راستای افقی حرکت می‌کند و دیگری در راستای عمودی و محاسبات گفته شده در بالا را برای هر کدام انجام می‌دهد.

کرنل scharr پیچیده تر از کرنل sobel است که به دلیل پیچیدگی های زیاد از توضیح آن صرف نظر کردیم ولی به طور کلی کرنل scharr به دلیل پیچیدگی های بیشتر دقت بالاتری خواهد داشت.

***در کد فقط از پارامتر gradient magnitude استفاده کردیم.

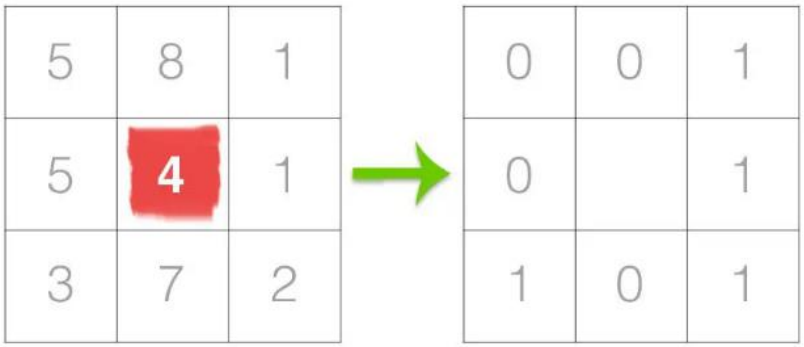
در نهایت با دو کرنل scharr و sobel دو ماتریس متشکل از gradient magnitude ها پیدا کردیم که با **میانگین و انحراف معیار و skew** آن را به عنوان فیچر استفاده کردیم.

Local Binary Pattern:

این ویژگی یک روش توصیف پیکسل های یک تصویر با توجه به پیکسل های همسایه آن است. در واقع این ویژگی هر پیکسل را با پیکسل های همسایه اش مقایسه می‌کند. این ویژگی دو شاخصه اصلی دارد:

- هزینه محاسباتی کم
- مقاومت در برابر نوسانات مقادیر gray scale(تغییرات شدید در روشنایی)

برای استخراج این ویژگی ابتدا باید عکس را grayscale کنیم و برای هر پیکسل تعداد مشخصی همسایه را انتخاب کنیم. الگوریتم استخراج این ویژگی در ابتدا به این شکل است که یک پیکسل و 8 همسایه اطراف آن را در نظر می‌گیرد(9 پیکسل) و عدد gray مربوط به هر پیکسل را با پیکسل مرکزی(threshold) مقایسه می‌کند. اگر بیشتر بود پیکسل لیبل یک می‌گیرد و اگر کمتر بود لیبل صفر می‌گیرد و به این ترتیب عکس به کد باینری تبدیل می‌شود.



در ادامه باید مقدار LBP را محاسبه کنیم. ماتریس 3*3 را در جهت ساعتگرد یا پادساعتگرد کنار هم قرار می‌دهیم و با فرمول زیر مقدار LBP را محاسبه میکنیم.

gc- *the intensity value of the central pixel*

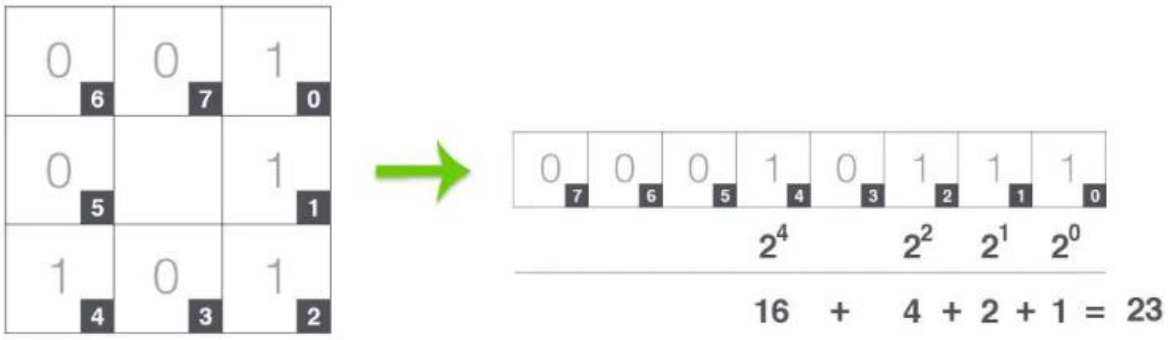
gp- *the intensity of the neighboring pixel with index p*

the function S can be expressed as:

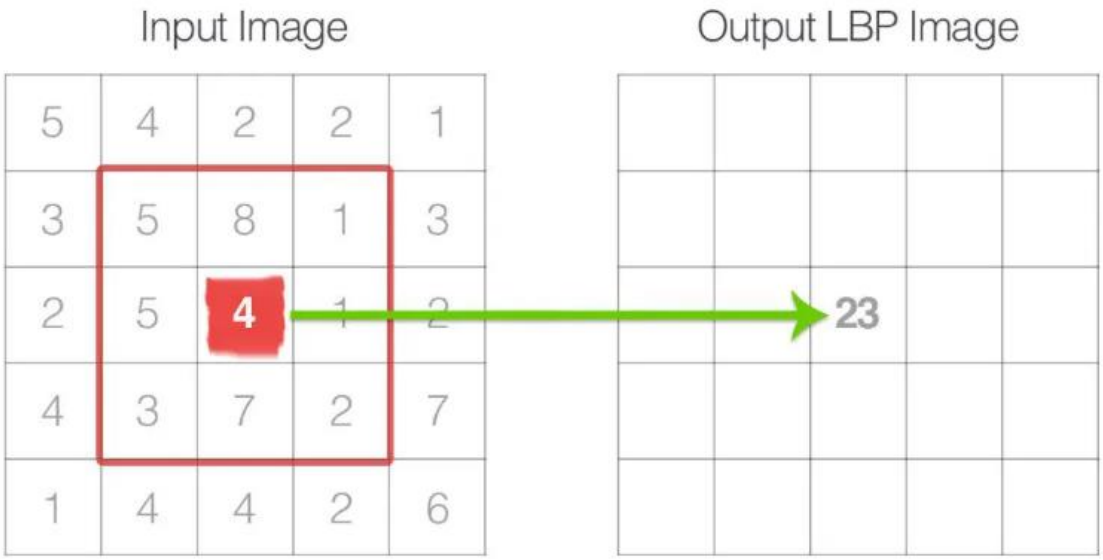
$$LBP(gp_x, gp_y) \sum_{p=0}^{P-1} S(gp - gc) \times 2^p$$

$$s(x) = \begin{cases} 1 & if \quad x \geq 0 \\ 0 & if \quad x < 0. \end{cases}$$

برای مثال در شکل زیر به صورت پادساعتگرد اعداد را در کنار هم قرار می‌دهیم و LBP را محاسبه می‌کنیم.

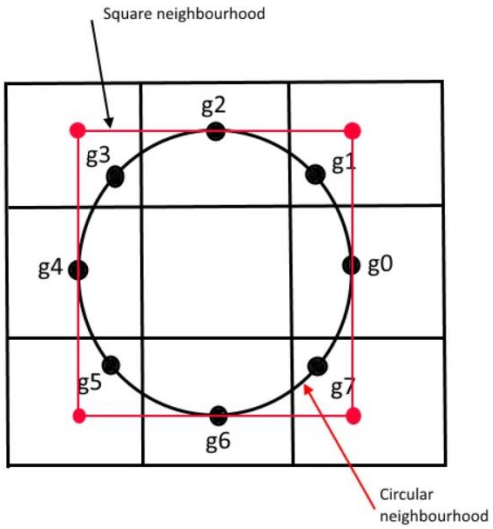


همین کار را برای ماتریس های 3*3 دیگر در کل عکس انجام میدهیم:



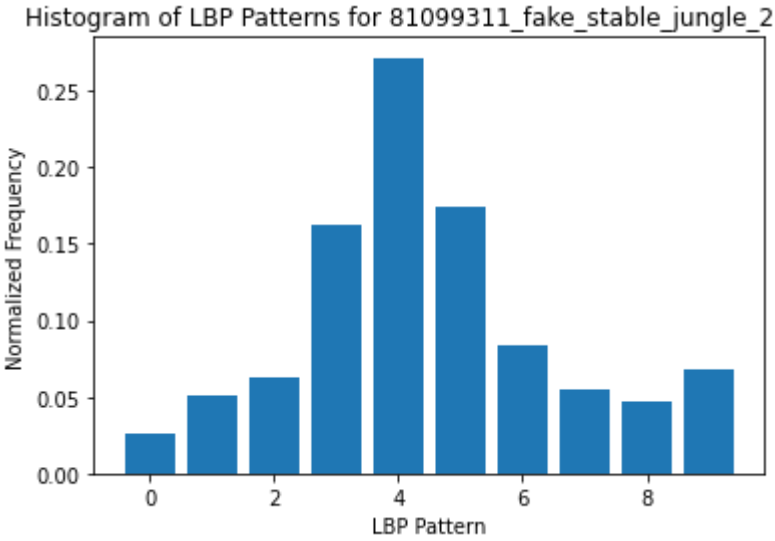
برای مثال عکس به صورت زیر تبدیل می شود:

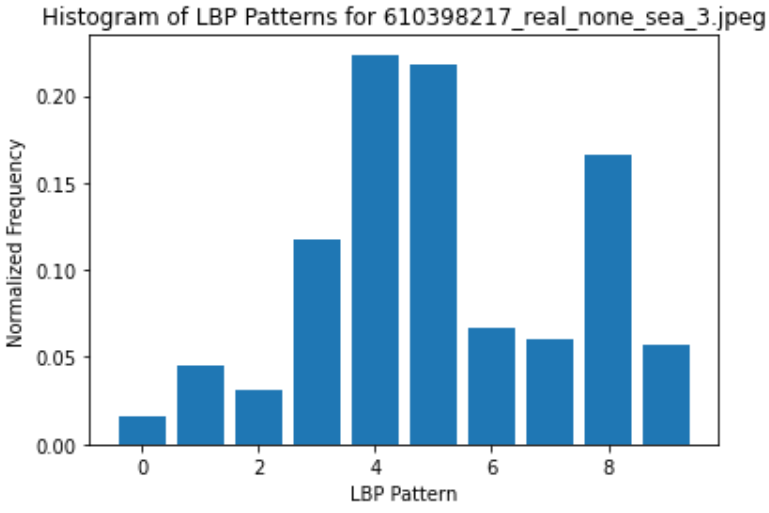
در الگوریتم های جدید دیگر ماتریس 3*3 در نظر گرفته نمی‌شود و یک دایره با شعاع مشخص به عنوان همسایگی در نظر گرفته می‌شود:



در نهایت هیستوگرام روی ماتریس LBP را بدست می‌آوریم این هیستوگرام نشانگر توزیع مربوط به اعداد روی ماتریس LBP است. ما از اعداد هیستوگرام به عنوان فیچر استفاده کردیم.

برای مثال برای دو تا از عکس های fake و real مراحل بالا را در کد پایتون انجام دادیم:






```
Features for image 1: [0.01556301 0.04509926 0.03141785 0.11694908 0.22360897 0.21754074
 0.06645775 0.06018925 0.16568565 0.05748844]
Features for image 2: [0.02547073 0.05094433 0.06214714 0.16288662 0.27164841 0.17356586
 0.08414745 0.05420494 0.04673004 0.06825447]
```

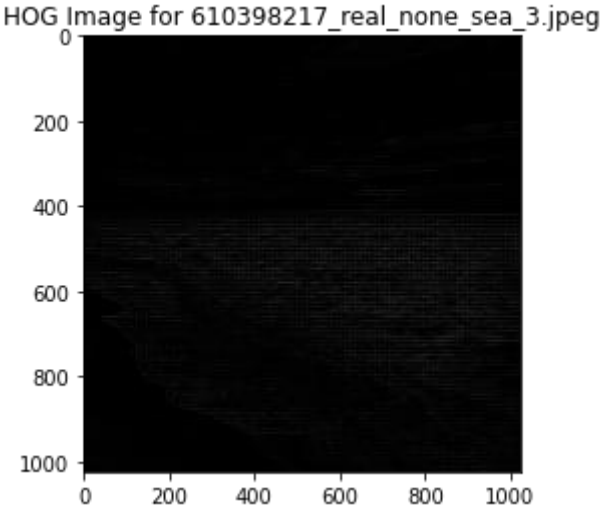
HOG features:

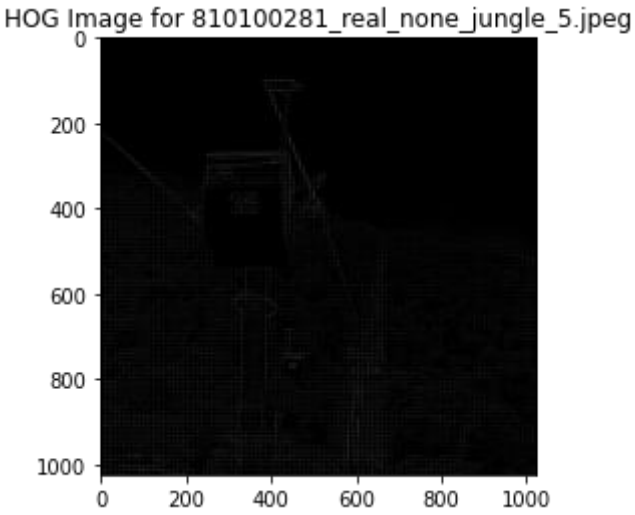
این الگوریتم جهت های لبه های موجود در تصاویر را استخراج می‌کند و این باعث می‌شود اجسام موجود در تصاویر شناسایی شوند. این کار با استخراج کردن گرادیان و جهت آنها در عکس انجام می‌شود. این الگوریتم تصویر را به بخش های مختلف تقسیم می‌کند و در هر بخش گرادیان و جهت آن را محاسبه می‌کند. بعد از محاسبه این دو این الگوریتم یک هیستوگرام برای هر ناحیه بدست می‌آید. به همین دلیل است که اسم این روش Histogram of Oriented Gradients می‌باشد.

در این ویژگی مشابه ویژگی luminance gradient گرادیان و زاویه مربوط به هر پیکسل را استخراج می‌شود و ماتریس گرادیان تشکیل می‌شود. سپس این الگوریتم با توجه به سایزی که مدنظر داریم این ماتریس را به ماتریس های n*n تقسیم می‌کند. برای مثال در کد ماتریس گرادیان را به ماتریس های 8*8 تقسیم کردیم. به این ماتریس ها بلوک می‌گوییم و برای هر بلوک یک هیستوگرام بدست می‌آوریم. سپس یک normalization روی هر بلوک انجام می‌دهیم و معمولا از L1-normalization یا L2-normalization استفاده می‌شود که در این پروژه از L2-normalization استفاده می‌کنیم.

بردار ویژگی نهایی از کنار هم قرار دادن هیستوگرام های نرمالیزه شده بدست می‌آید. در پروژه برای هر عکس 34 فیچر بدست آمد که فقط میانگین و انحراف معیارهای آنها را در نظر گرفتیم.

نتایج خروجی:





اگر در عکس های فوق دقت کنیم می‌بینیم که edge های موجود در تصاویر و object های آن تا حدی تشخیص داده شده اند.

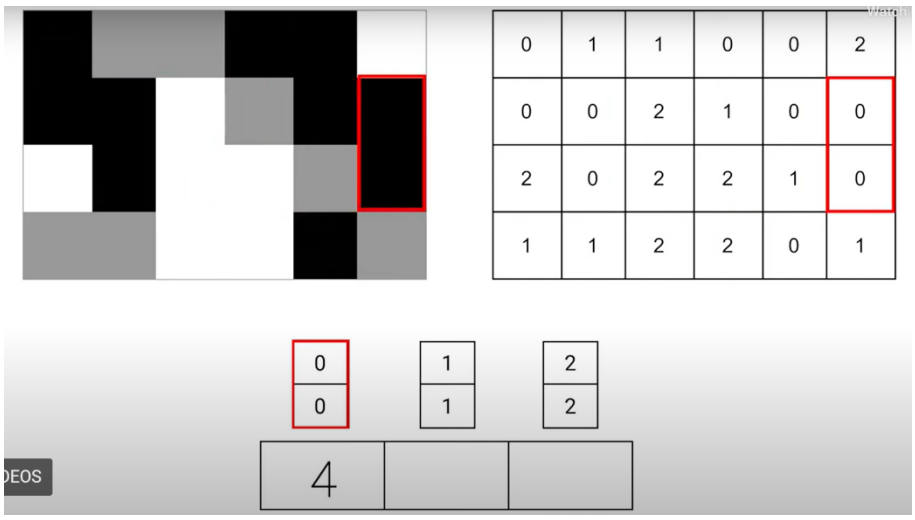
Gray-Level Co-occurrence Matrix (GLCM):

این ویژگی که یکی از مهم ترین ویژگی هایی بود که استخراج کردیم اطلاعاتی درباره وابستگی مکانی پیکسل ها به یکدیگر را می‌دهد. این الگوریتم برای محاسبه پارامترهای مختلف مراحل زیر را طی می‌کند:

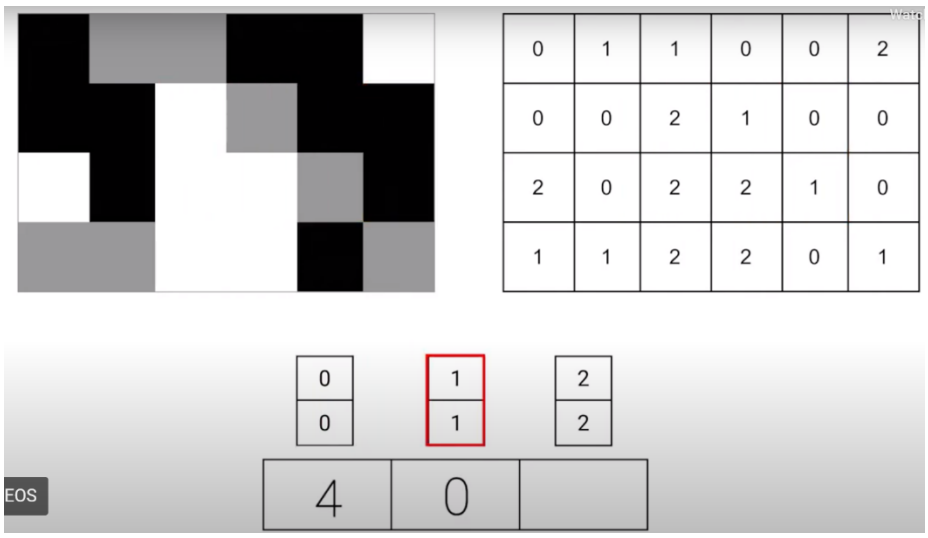
- ابتدا عکس ها را gray می‌کنیم و به این ترتیب برای هر پیکسل یک gray-level بدست می‌آید.
- بدست آوردن ماتریس Co-occurrence Matrix:

این الگوریتم بررسی می‌کند که پیکسل i ام با gray-level مشخص چند بار در نزدیکی پیکسل j ام در عکس بوده است.

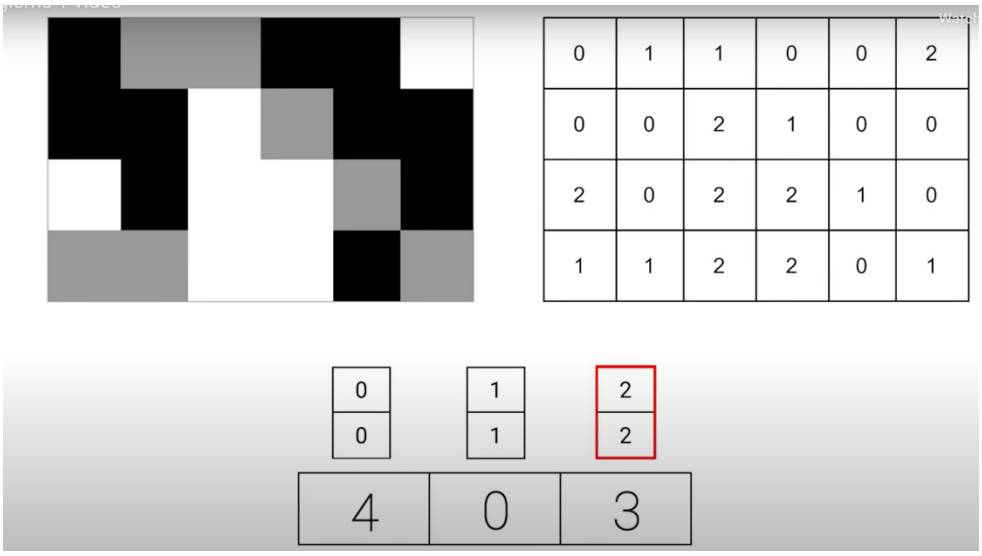
پیکسل های (0,0):



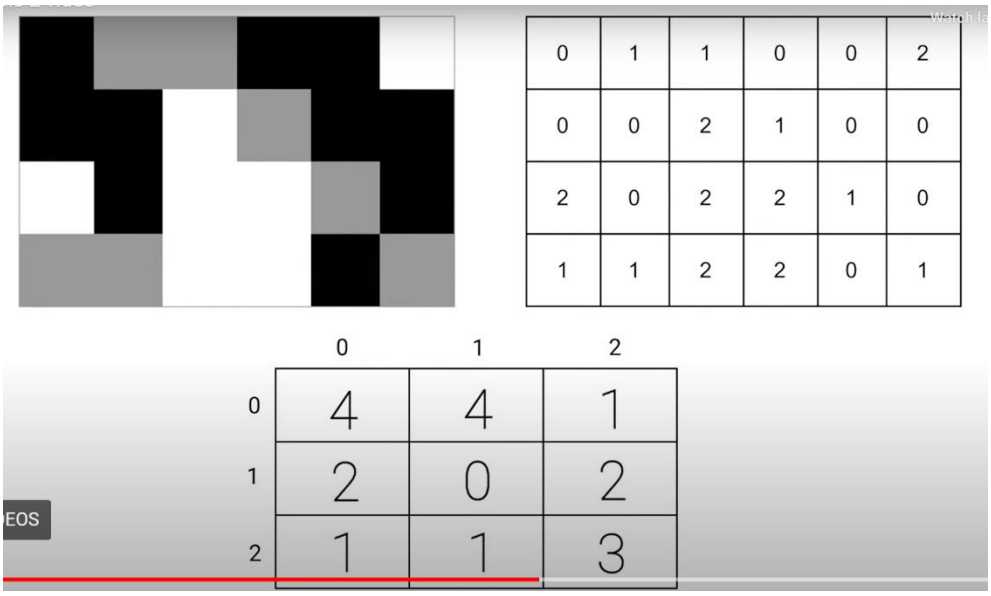
پیکسل های (1,1):



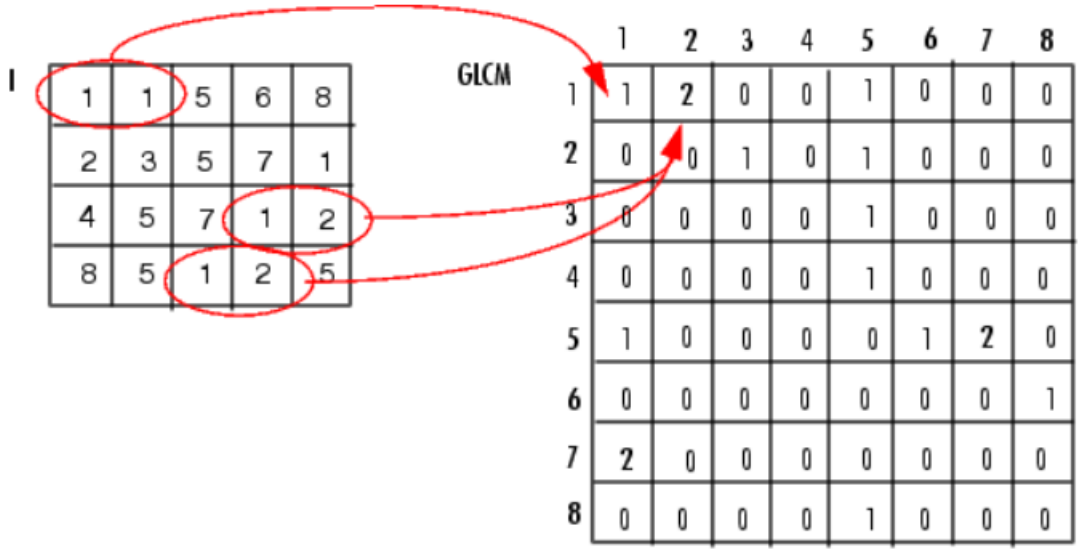
پیکسل های(2,2):



اگر همین کار را برای ترکیب دو دویی سایر پیکسل ها تکرار کنیم ماتریسی به شکل زیر بدست می‌آید:



چون در عکس های واقعی gray-level ها متنوع تر هستن ابعاد این ماتریس بزرگ تر می‌شود:



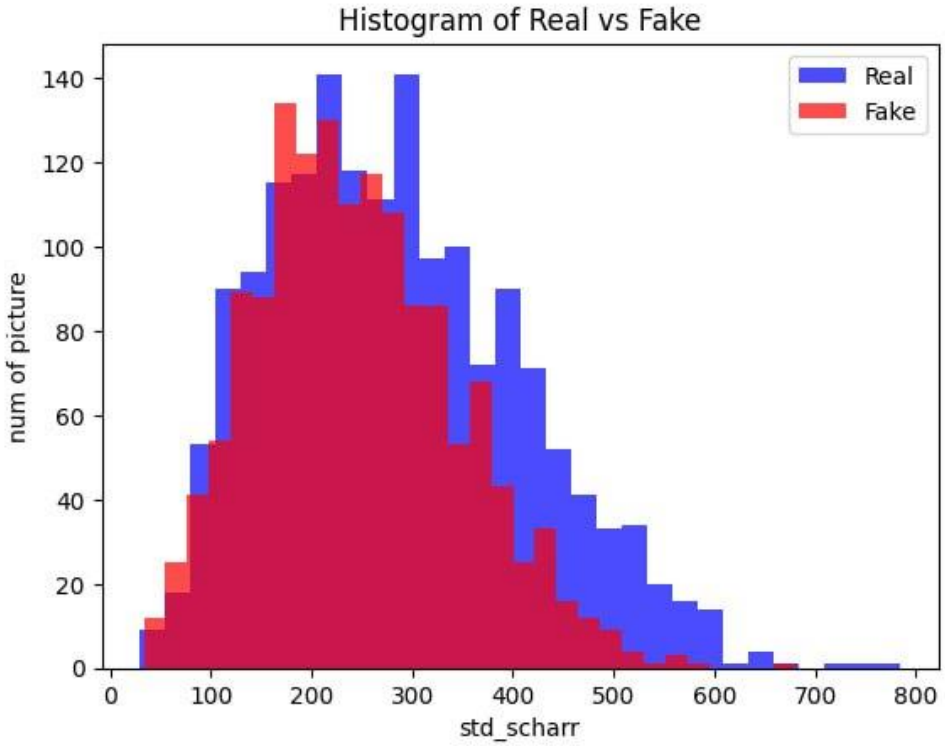
بعد از بدست آوردن این ماتریس که حاوی اطلاعات مکانی پکسل ها است می‌توانیم پارامتر های مختلف را محاسبه کنیم:

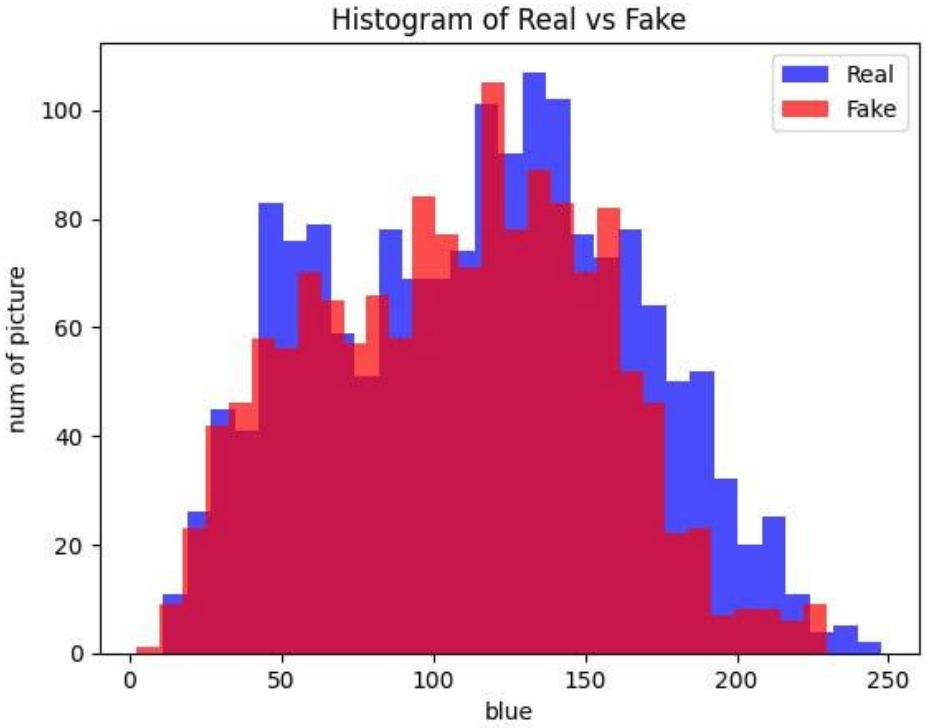
- 'contrast': $\sum_{i,j=0}^{levels-1} P_{i,j}(i-j)^2$
- 'dissimilarity': $\sum_{i,j=0}^{levels-1} P_{i,j}|i-j|$
- 'homogeneity': $\sum_{i,j=0}^{levels-1} \frac{P_{i,j}}{1+(i-j)^2}$
- 'ASM': $\sum_{i,j=0}^{levels-1} P_{i,j}^2$
- 'energy': $\sqrt{ASM}$
- **'correlation':**

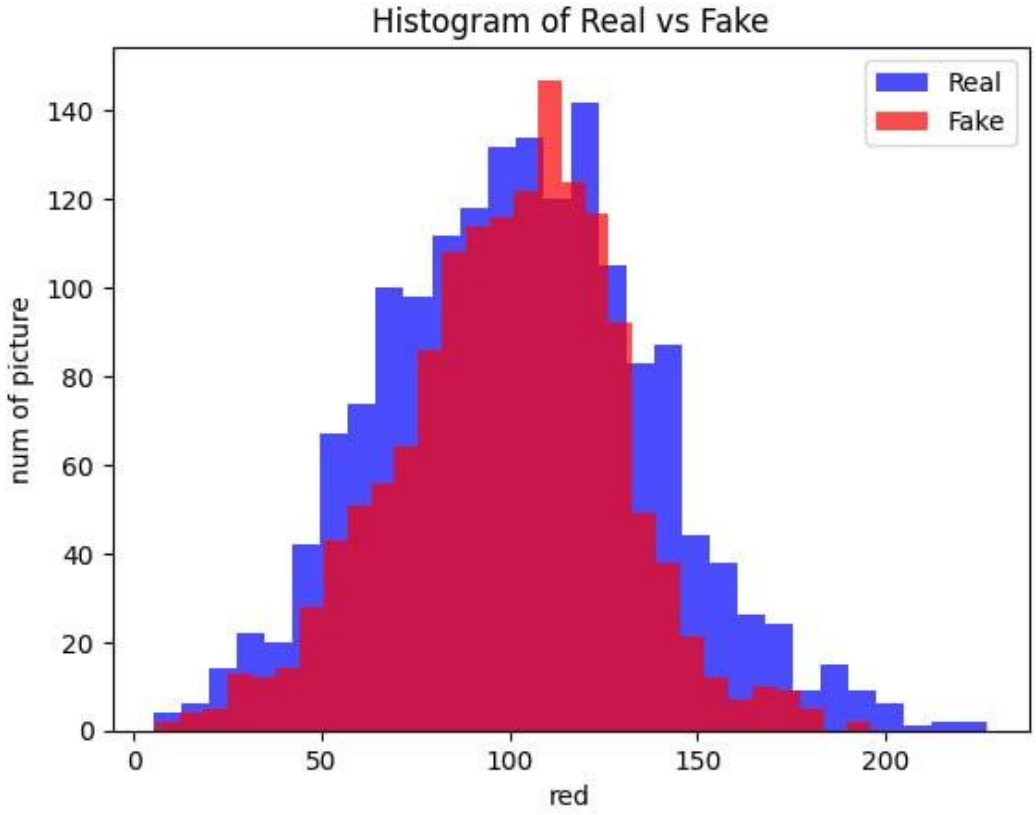
$$\sum_{i,j=0}^{levels-1} P_{i,j} \left[\frac{(i-\mu_i)\,(j-\mu_j)}{\sqrt{(\sigma_i^2)(\sigma_j^2)}} \right]$$

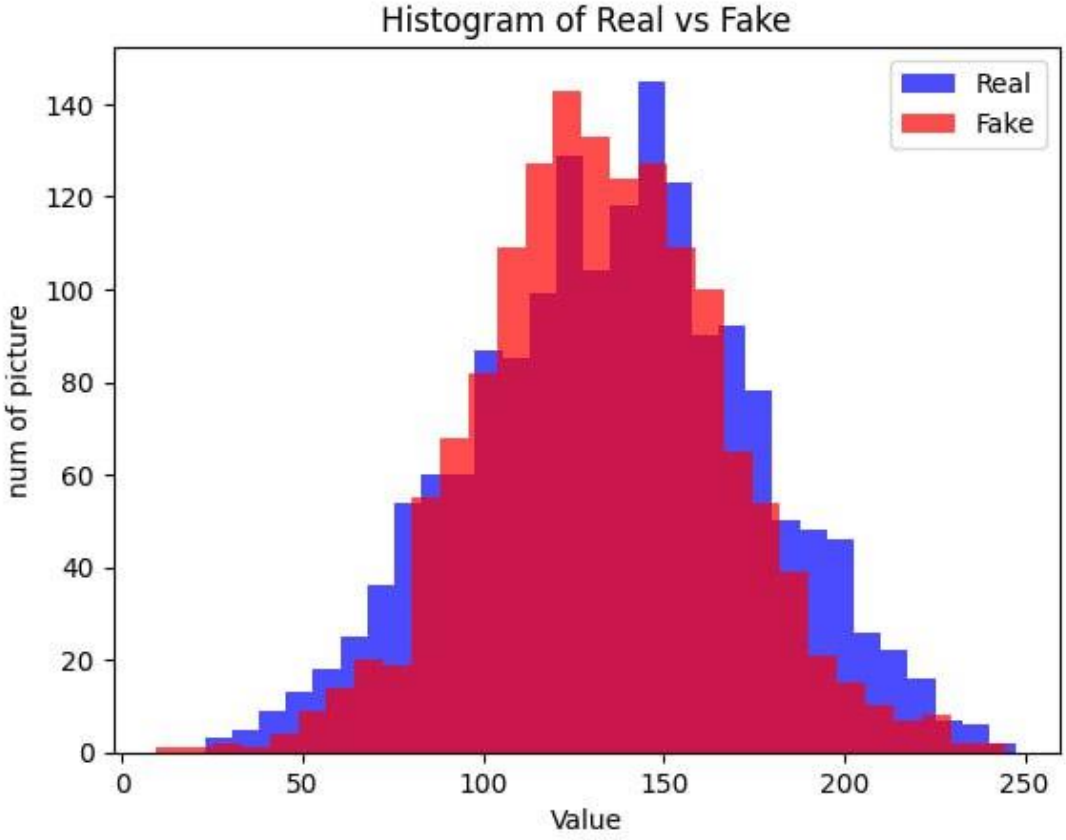
نکته درباره ی Metadata : با استفاده از Metadata هر عکس، میتوانیم ویژگی ای برای تشخیص واقعی یا ساختگی بودن عکس استخراج کنیم . اما از بعضی از عکسها نتوانستیم متا دیتا استخراج کنیم بنابراین از متادیتا برای طبقه بندی و به عنوان فیچر استفاده نکردیم .

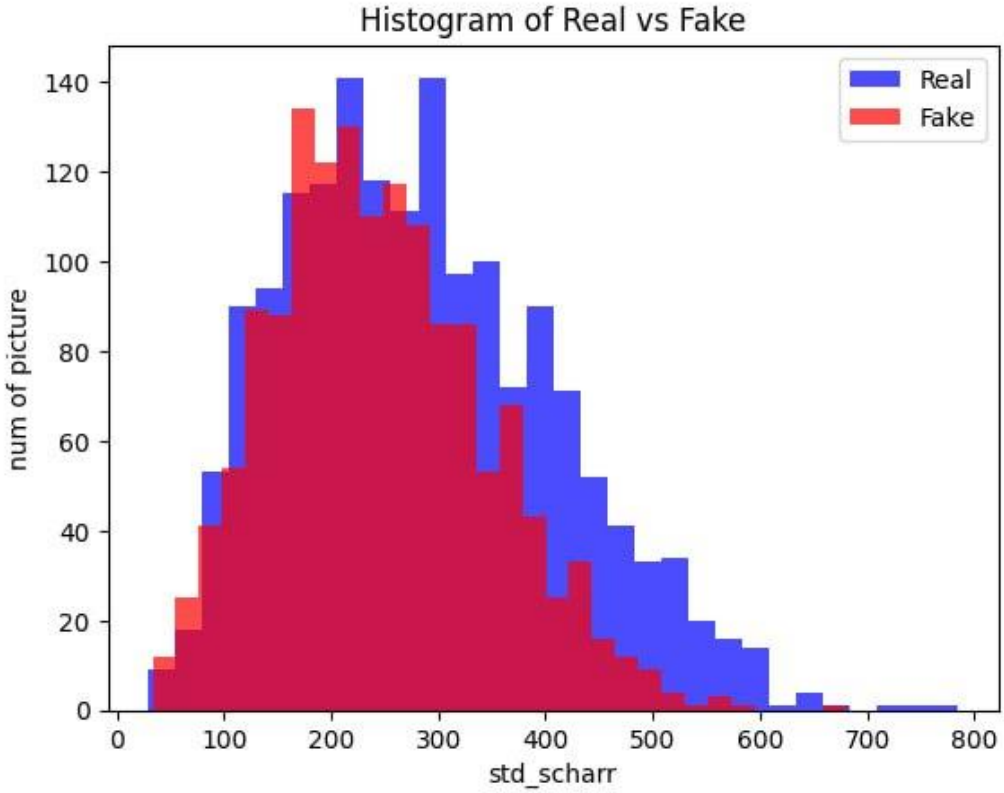
در ادامه توزیع هیستوگرام برخی ویژگی ها را مشاهده میکنیم :

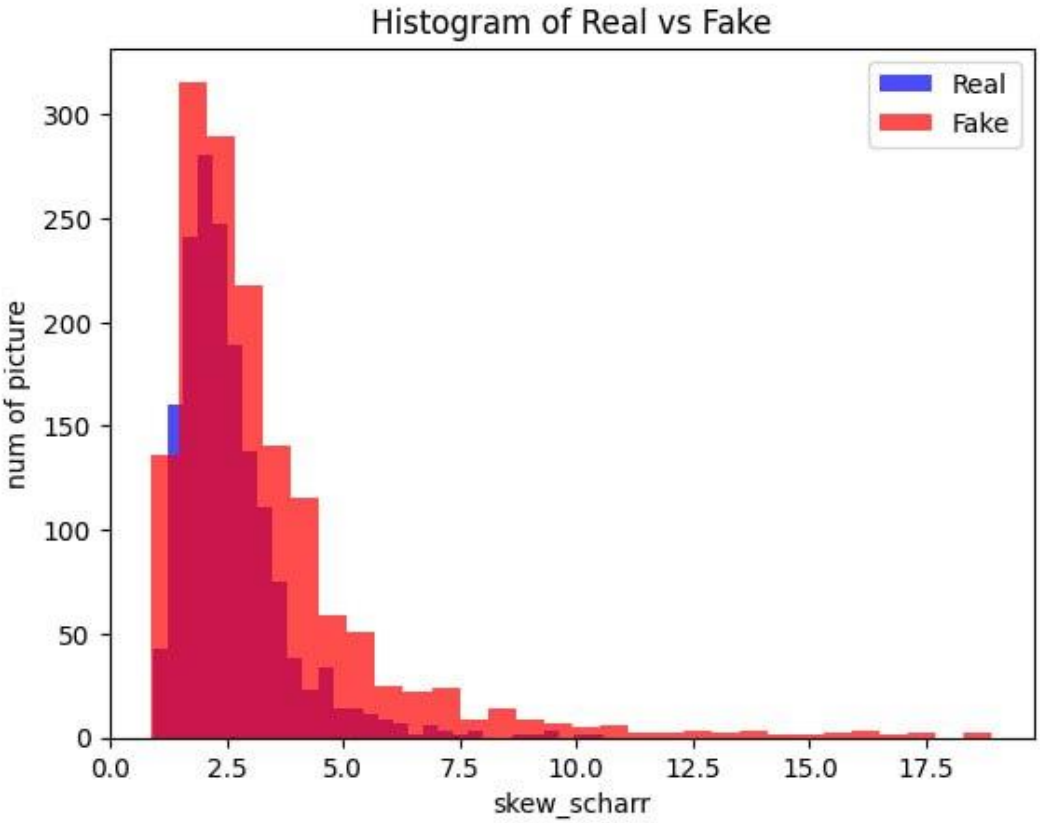


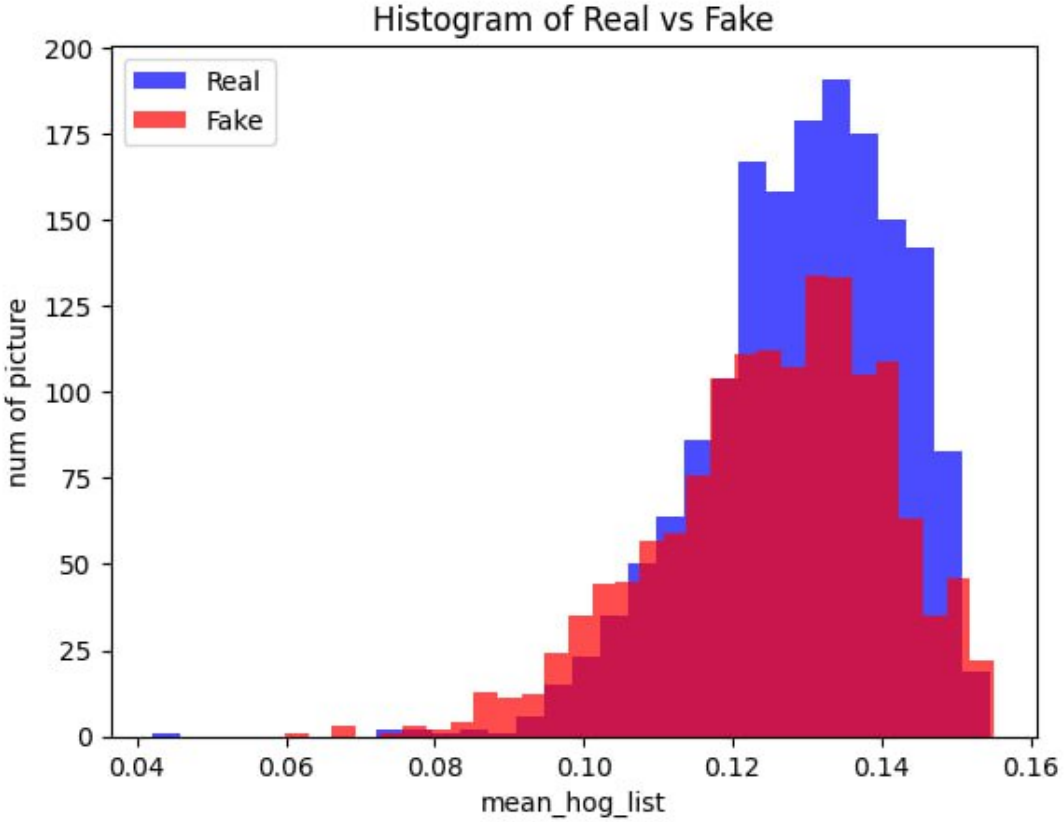


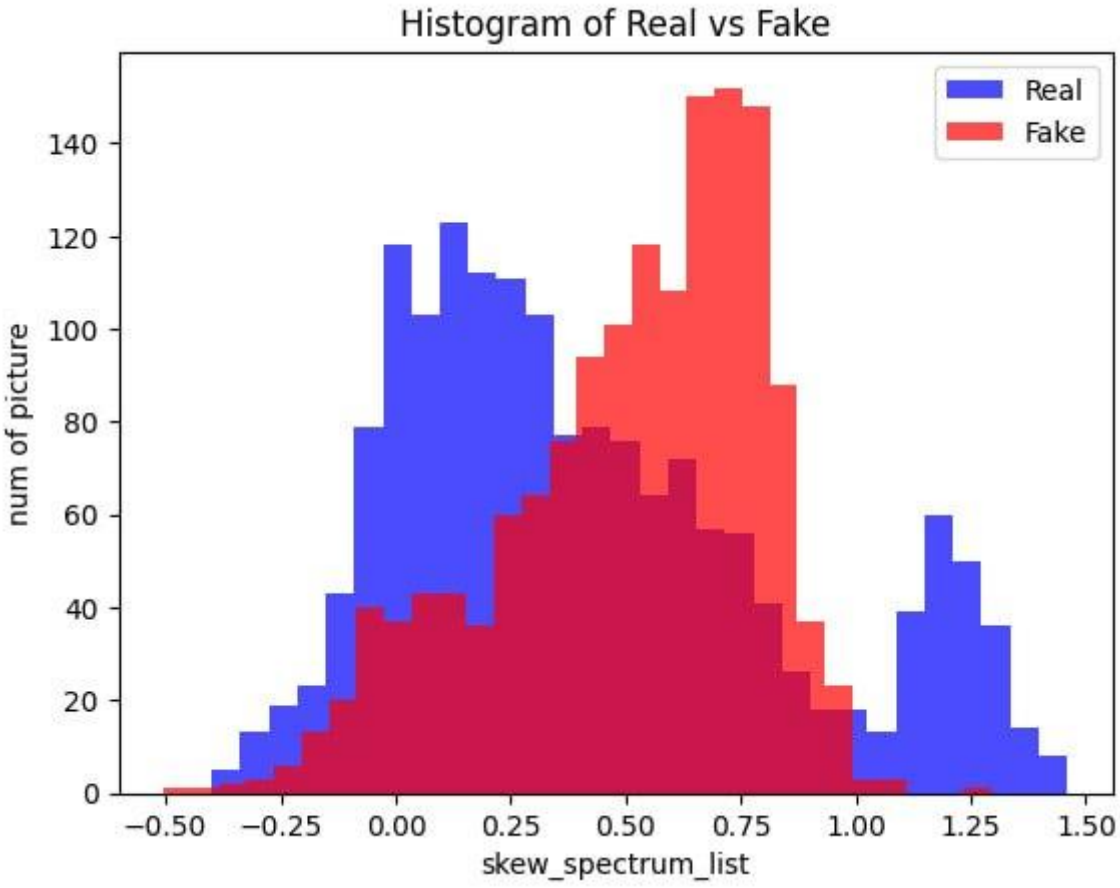

Histogram of Real vs Fake
Real
Fake
num of picture
red
0
20
40
60
80
100
120
140
0
50
100
150
200

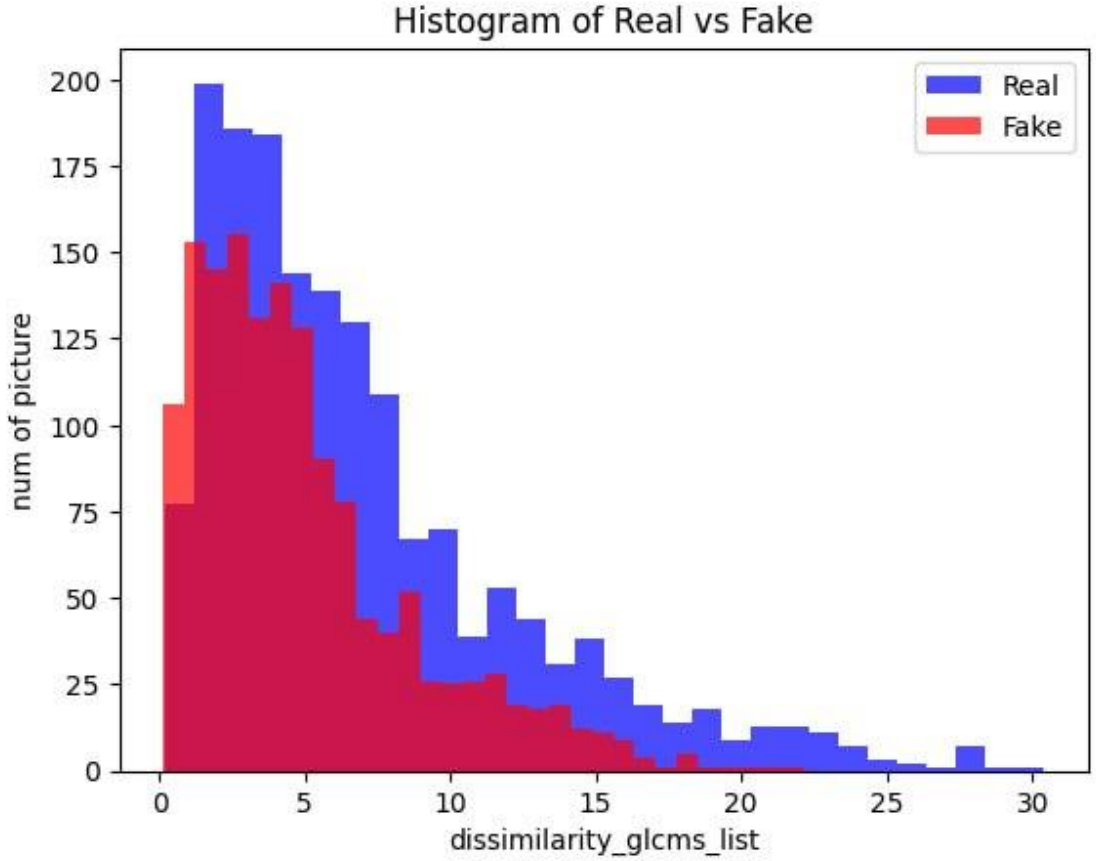

Histogram of Real vs Fake
Real
Fake
num of picture
Value
0
20
40
60
80
100
120
140
0
50
100
150
200
250

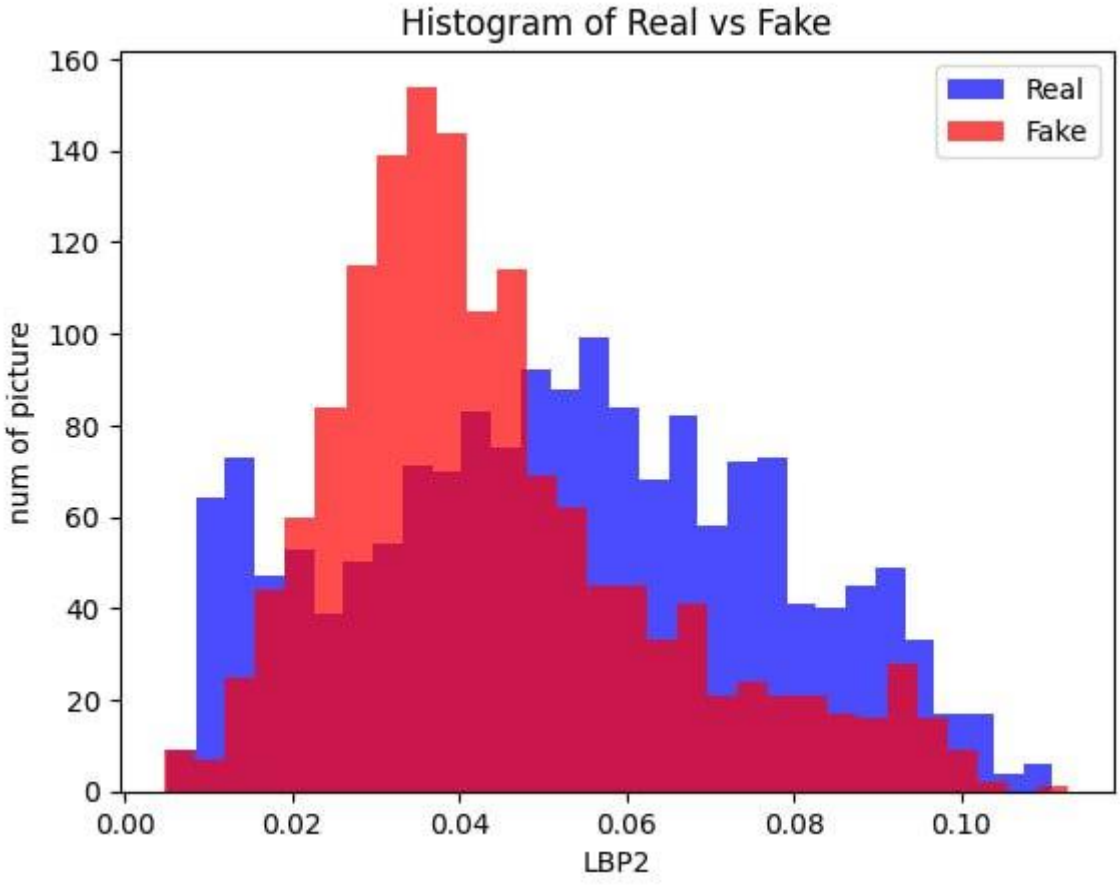

Histogram of Real vs Fake
Real
Fake
num of picture
std_scharr
0
20
40
60
80
100
120
140
0
100
200
300
400
500
600
700
800

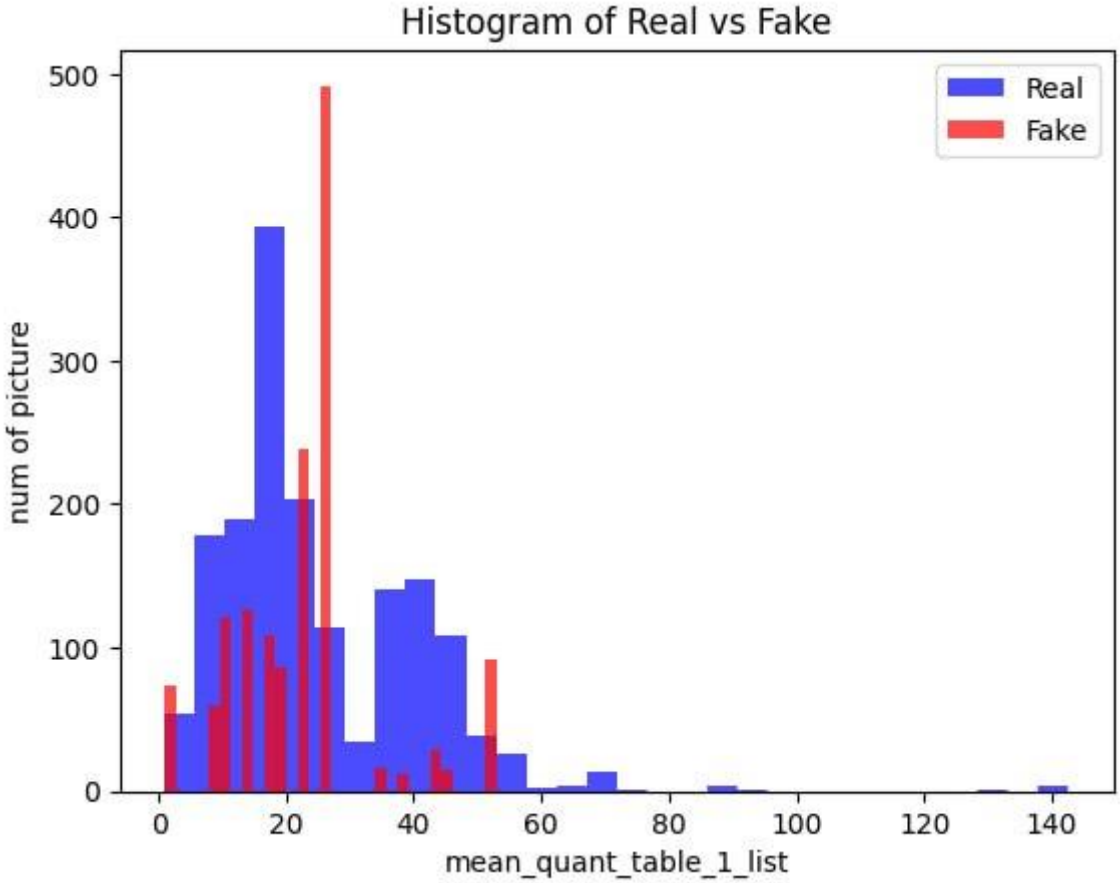

Histogram of Real vs Fake
Real
Fake
num of picture
skew_scharr
0
50
100
150
200
250
300
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5

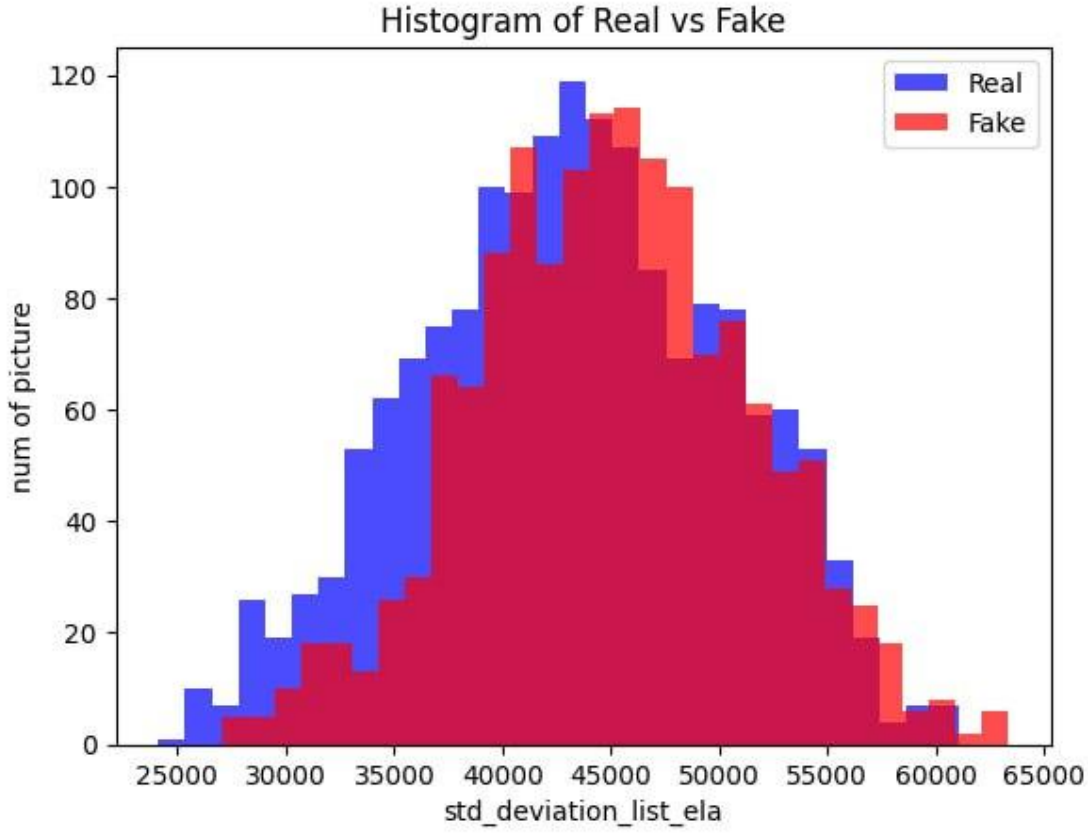

Histogram of Real vs Fake
Real
Fake
num of picture
mean_hog_list
0
25
50
75
100
125
150
175
200
0.04
0.06
0.08
0.10
0.12
0.14
0.16


Histogram of Real vs Fake
Real
Fake
num of picture
skew_spectrum_list
0
20
40
60
80
100
120
140
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00
1.25
1.50


Histogram of Real vs Fake
Real
Fake
num of picture
dissimilarity_glcms_list
0
25
50
75
100
125
150
175
200
0
5
10
15
20
25
30


Histogram of Real vs Fake
Real
Fake
num of picture
LBP2
0
20
40
60
80
100
120
140
160
0.00
0.02
0.04
0.06
0.08
0.10


Histogram of Real vs Fake
Real
Fake
num of picture
mean_quant_table_1_list
0
100
200
300
400
500
0
20
40
60
80
100
120
140


Histogram of Real vs Fake
Real
Fake
num of picture
std_deviation_list_ela
0
20
40
60
80
100
120
25000
30000
35000
40000
45000
50000
55000
60000
65000

سوال سوم: طبقه بندی

برای طبقه بندی ، داده ها را با نسبت 70 به 30 ، به داده های آموزشی و تست تقسیم میکنیم با استفاده از دستور train_test_split.

طبقه بندی با استفاده از داده های استخراج شده ی خودمان

طبقه بندی با استفاده از درخت تصمیم

چرایی انتخاب درخت تصمیم برای طبقه بندی:

1. تفسیرپذیری: درختان تصمیم بسیار قابل تفسیر هستند و درک روشنی از نحوه تصمیم گیری طبقه بندی ارائه می دهند. ساختار درختی را می توان به راحتی تجسم و درک کرد، که آن را برای توضیح استدلال پشت پیش بینی های طبقه بندی مفید می کند.

2. روابط غیر خطی: درخت های تصمیم می توانند روابط غیرخطی بین ویژگی ها و متغیر هدف را ثبت کنند. آنها می توانند مرزهای تصمیم گیری پیچیده را کنترل کنند و مانند برخی دیگر از طبقه بندی کننده ها به تفکیک پذیری خطی محدود نمی شوند.

3. اهمیت ویژگی: درختان تصمیم معیاری از اهمیت ویژگی را ارائه می دهند که نشان دهنده اهمیت نسبی ویژگی های مختلف در تصمیم گیری های طبقه بندی است. این اطلاعات می تواند برای انتخاب ویژگی و درک تأثیر ویژگی ها بر نتیجه طبقه بندی مفید باشد.

4. پیاده سازی آسان: پیاده سازی و درک درختان تصمیم نسبتاً آسان است. کتابخانه‌ها و بسته‌های به‌راحتی در زبان‌های برنامه‌نویسی مختلف وجود دارند که پیاده‌سازی درخت تصمیم را فراهم می‌کنند و استفاده از آنها را برای کارهای طبقه‌بندی راحت می‌کنند.

علیرغم این مزایا، توجه به این نکته مهم است که درختان تصمیم نیز محدودیت هایی دارند. آنها می توانند مستعد overfit باشند، به خصوص زمانی که درخت عمیق رشد می کند یا هنگام مواجهه با مجموعه داده های نویزی یا نامتعادل. با این حال، این محدودیت‌ها را می‌توان با استفاده از تکنیک‌هایی مانند هرس، روش‌های گروهی و منظم‌سازی کاهش داد.

ارزیابی مدل :

```
Confusion Matrix:
[[399 112]
 [ 91 417]]

Accuracy: 0.8007850834151129
Error: 0.19921491658488713
Precision: 0.7882797731568998
Recall: 0.8208661417322834
F1-Score: 0.8042430086788813
```

Figure 4 ارزیابی مدل درخت تصمیم

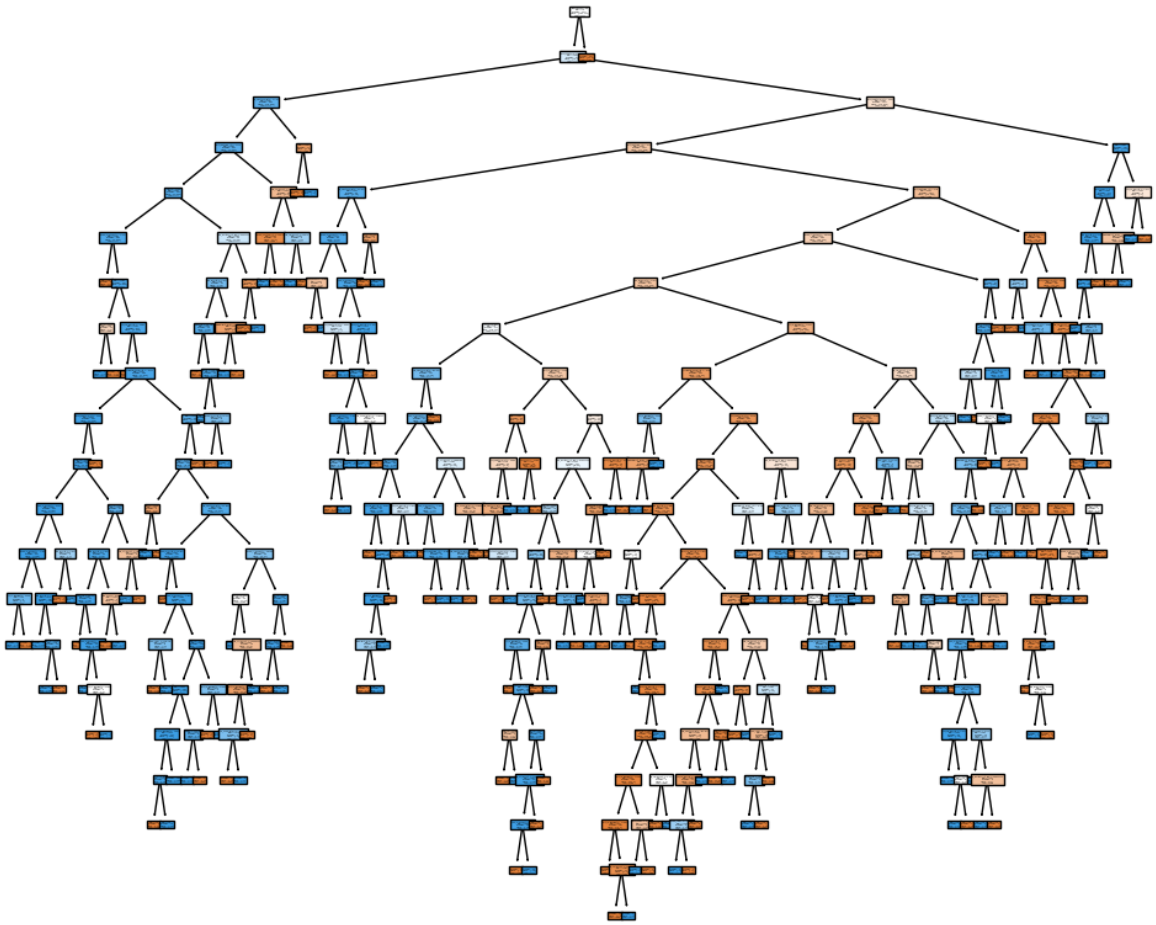


Figure 5 نمایی از مدل درخت تصمیم

طبقه بندی با استفاده از درخت تصمیم(max depth = 3)

ارزیابی مدل :

```
Confusion Matrix:
[[464  47]
 [198 310]]

Accuracy: 0.7595682041216879
Error: 0.24043179587831207
Precision: 0.8683473389355743
Recall: 0.610236220472441
F1-Score: 0.7167630057803469
```

Figure 6 ارزیابی مدل درخت تصمیم(**max depth = 3**)

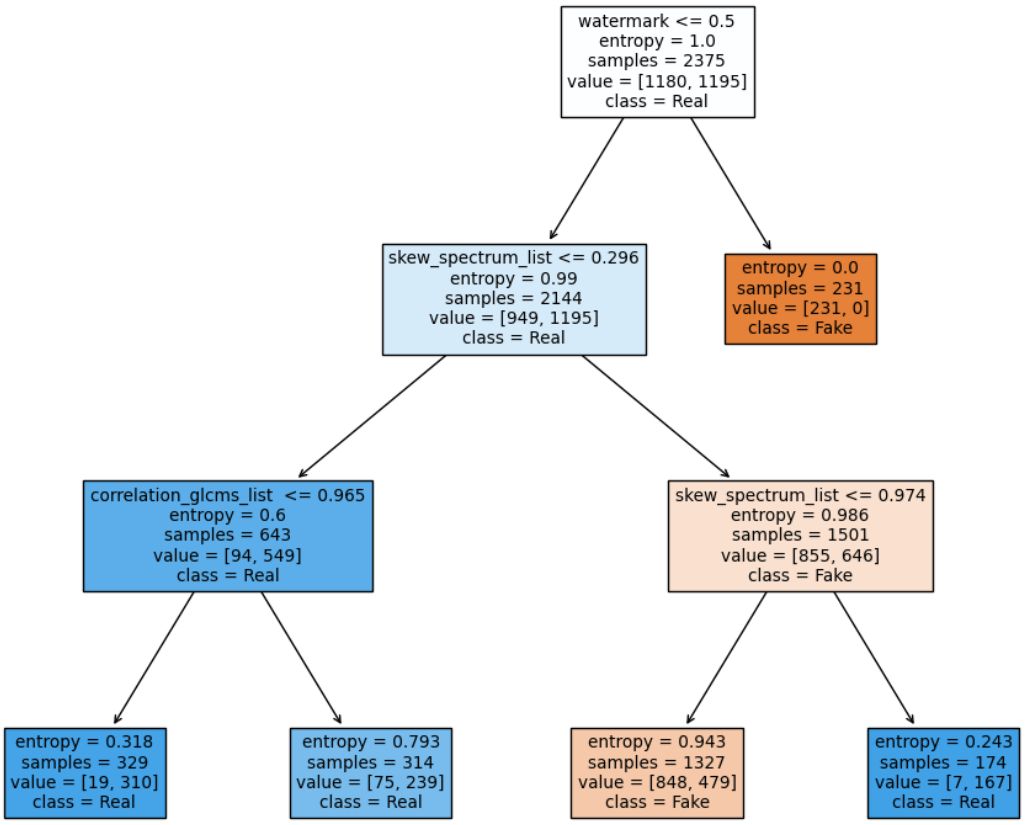


Figure 7 نمایی از مدل درخت تصمیم

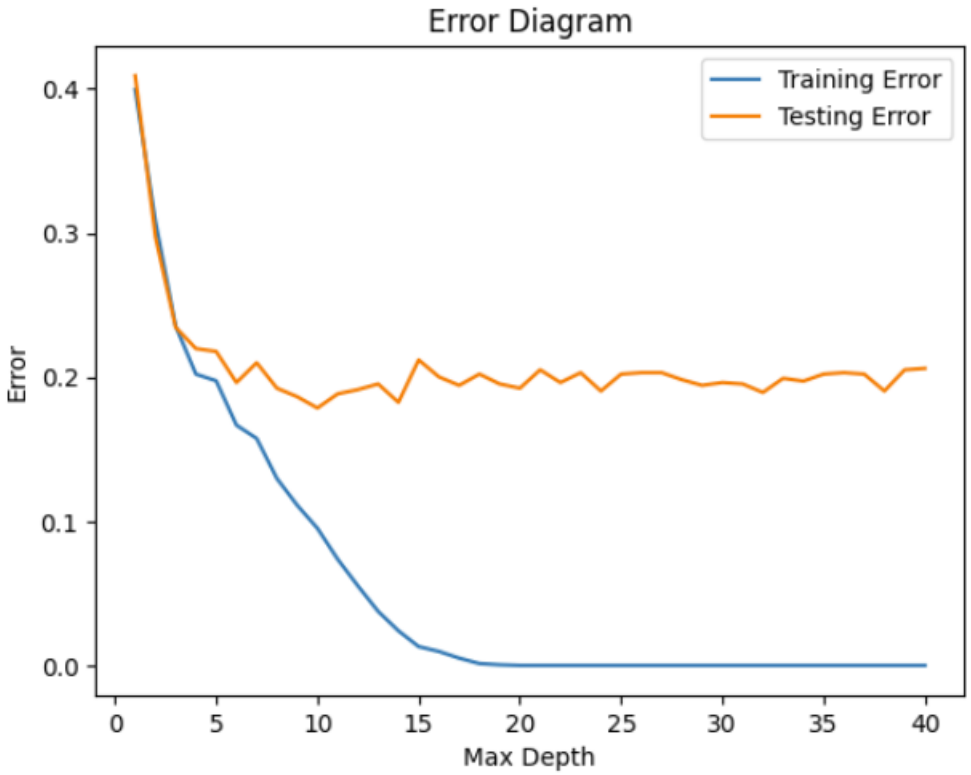


Figure8 Error diagram for different value of max depth

همانطور که مشاهده می شود با افزایش عمق درخت ، خطای آموزش کم می شود . اما خطای تست به یک مینیمم می رسد و بعد شروع به افزایش میکند . بنابراین میتوان عمق بهینه را پیدا کرد .

طبقه بندی با استفاده از طبقه بند SVM

چرایی انتخاب SVM برای طبقه بندی:

1. موثر در فضاهای با ابعاد بالا: SVM ها در فضاهای با ابعاد بالا عملکرد خوبی دارند و آنها را برای مشکلات با تعداد زیادی ویژگی مناسب می کند. آنها می توانند مجموعه داده هایی با هزاران بعد را مدیریت کنند و عملکرد تعمیم خوبی داشته باشند.

2. قابلیت تعمیم بالا: هدف SVM ها یافتن یک مرز تصمیم گیری بهینه است که حاشیه بین کلاس ها را به حداکثر می رساند. این رویکرد به حداکثر رساندن حاشیه منجر به توانایی تعمیم خوب می شود، به این معنی که SVM ها اغلب روی داده های دیده نشده عملکرد خوبی دارند.

3. تطبیق پذیری در توابع هسته: SVM ها می توانند از کرنل های مختلف (به عنوان مثال کرنل خطی(linear)، چند جمله ای(polynomial)، شعاعی(rbf)) برای تبدیل داده های ورودی به فضایی با ابعاد بالاتر استفاده کنند. این انعطاف پذیری به SVM ها اجازه می دهد تا مرزهای تصمیم گیری پیچیده را مدیریت کنند و روابط غیرخطی داده ها را پیدا کنند.

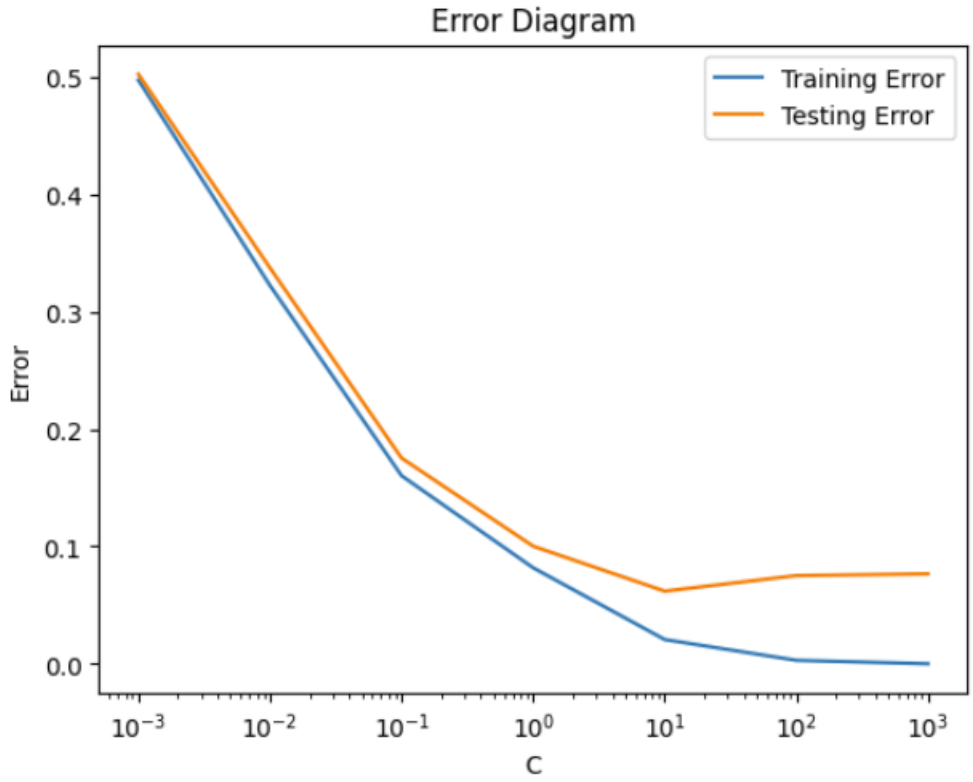
4. مقاوم به بیش برازش : SVMها کمتر مستعد overfitting هستند، به خصوص در سناریوهایی با نمونه های آموزشی محدود. در SVM چون به دنبال حد اکثر کردن حاشیه هستیم ، از overfitting جلوگیری می شود .

5. Regularization: SVM ها دارای یک پارامتر منظم سازی (C) هستند که امکان کنترل trade-off بین حداکثر کردن حاشیه و به حداقل رساندن خطاهای طبقه بندی را فراهم می کند. این پارامتر به تنظیم پیچیدگی مدل و مدیریت trade-off خطای بایاس واریانس کمک می کند.

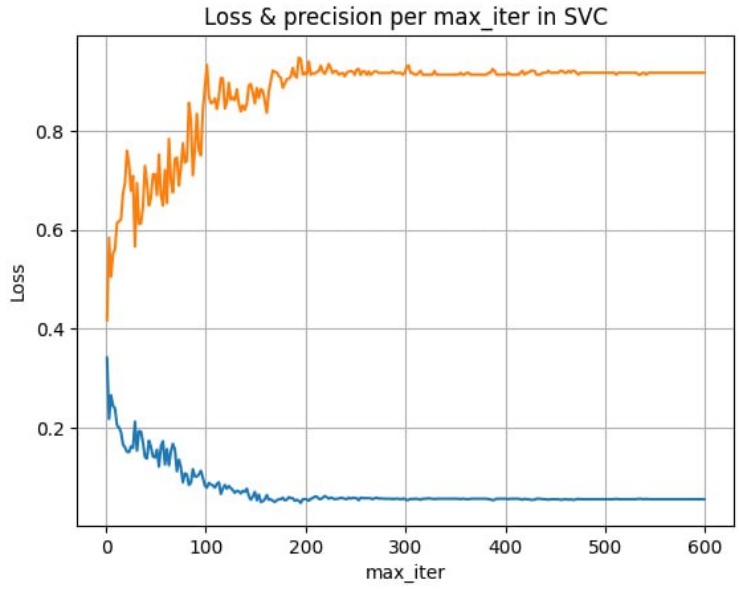
ارزیابی مدل :

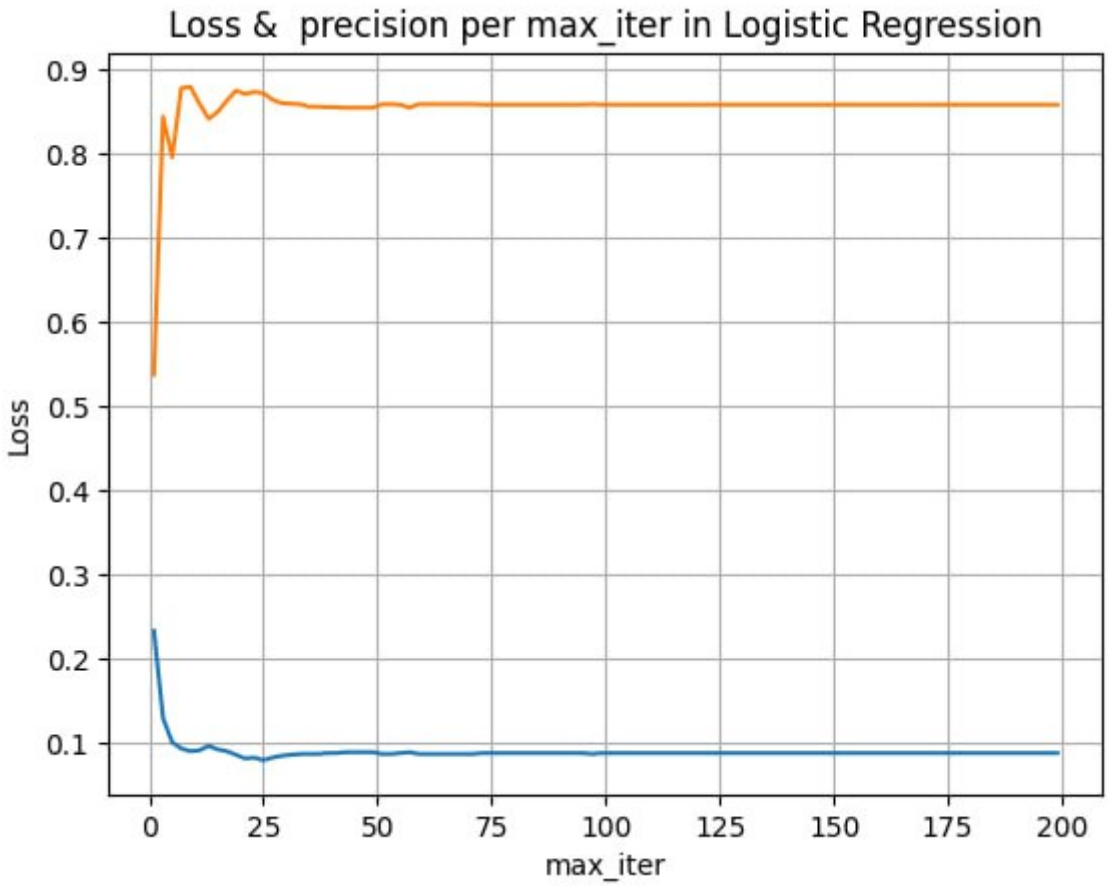
```
Confusion Matrix:
[[317  24]
 [ 44 294]]

Accuracy: 0.8998527245949927
Error: 0.10014727540500737
Precision: 0.9245283018867925
Recall: 0.8698224852071006
F1-Score: 0.8963414634146342
```

Figure 9 ارزیابی مدل **SVM**



در SVM (ماشین بردار پشتیبانی)، پارامتر C یک پارامتر منظم‌سازی(regularization) است که مبادله(trade-off) بین حداکثر کردن حاشیه و به حداقل رساندن خطای طبقه‌بندی را کنترل می‌کند. این نرمی یا سختی حاشیه را تحت تأثیر قرار می دهد و بر توانایی SVM برای طبقه بندی صحیح نمونه های آموزشی تأثیر می گذارد. در اینجا نیز با تغییر دادن پارامتر C تاثیرش را در خطای تست و آموزش میبینیم.





Confusion matrix :

```
[[652  34]
 [ 49 204]]
```

طبقه بندی با استفاده از LDA

چرایی انتخاب LDA برای طبقه بندی:

1. کاهش ابعاد: LDA ابعاد فضای ویژگی را کاهش می دهد در حالی که ویژگی هایی که باعث تفکیک پذیری بین کلاس ها می شود را ، حفظ می کند. LDA داده ها را بر روی یک زیرفضای با ابعاد پایین تر پخش می کند و قابلیت تفکیک کلاسها را به حداکثر می رساند. با کاهش تعداد ویژگی‌ها، LDA می‌تواند مشکل طبقه‌بندی را ساده‌سازی کند، کارایی محاسباتی را بهبود بخشد و خطر overfitting را کاهش دهد.

2. مرز تصمیم بهینه: هدف LDA یافتن یک مرز تصمیم بهینه است که تفکیک کلاسها را به حداکثر برساند. با به حداکثر رساندن نسبت پراکندگی بین کلاسی به پراکندگی درون کلاسی به این امر دست می یابد. این منجر به یک مرز تصمیم می شود که همپوشانی بین کلاس ها را به حداقل می رساند و دقت طبقه بندی را بهبود می بخشد.

3. پیش‌بینی کلاس: از LDA می‌توان برای پیش‌بینی کلاس با تخصیص نقاط داده جدید به کلاس با نزدیک‌ترین میانگین یا بر اساس مرز تصمیم‌گیری استفاده کرد.

4. مدل خطی: LDA خطی بودن داده ها را فرض می کند و زمانی که کلاس ها به خوبی با مرزهای خطی از هم جداپذیر باشند، به خوبی کار می کند.

ارزیابی مدل :

```
Confusion Matrix:
[[310  31]
 [ 40 298]]

Accuracy: 0.8954344624447718
Error: 0.10456553755522828
Precision: 0.9057750759878419
Recall: 0.8816568047337278
F1-Score: 0.8935532233883058
```

Figure 10 ارزیابی **LDA**

طبقه بندی با استفاده از کاهش بعد به روش PCA و سپس استفاده از طبقه بند SVM

ارزیابی مدل :

```
Confusion Matrix:
[[311  30]
 [ 57 281]]

Accuracy: 0.8718703976435935
Error: 0.12812960235640647
Precision: 0.9035369774919614
Recall: 0.8313609467455622
F1-Score: 0.8659476117103235
```

Figure 11 ارزیابی مدل **SVM** بعد از کاهش بعد به 35 بعد

مقایسه طبقه بند ها : طبق خروجی های بالا ، طبقه بند SVM بهترین طبقه بند (با دقت بیشتر) است.

در ادامه ، از داده هایی که توسط دستیار آموزشی استخراج شده بود ، و با استفاده از طبقه بند درخت تصمیم ، میاییم داده ها را طبقه بندی میکنیم .

طبقه بندی با استفاده از داده های استخراج شده توسط دستیار آموزشی

طبقه بندی با استفاده از درخت تصمیم

ارزیابی مدل :

```
Confusion Matrix:
[[516   3]
 [  1 506]]

Accuracy: 0.9961013645224172
Error: 0.003898635477582846
Precision: 0.9941060903732809
Recall: 0.9980276134122288
F1-Score: 0.9960629921259843
```

Figure 12 ارزیابی مدل درخت تصمیم

در اثر این طبقه بندی ، 4 تا سمپل از سمپل های تست ، اشتباه طبقه بندی می شوند(دقت طبقه بندی برابر با 99.61درصد شد) که آن 4 تا به شرح زیر است:

Figure13 fake but detected real

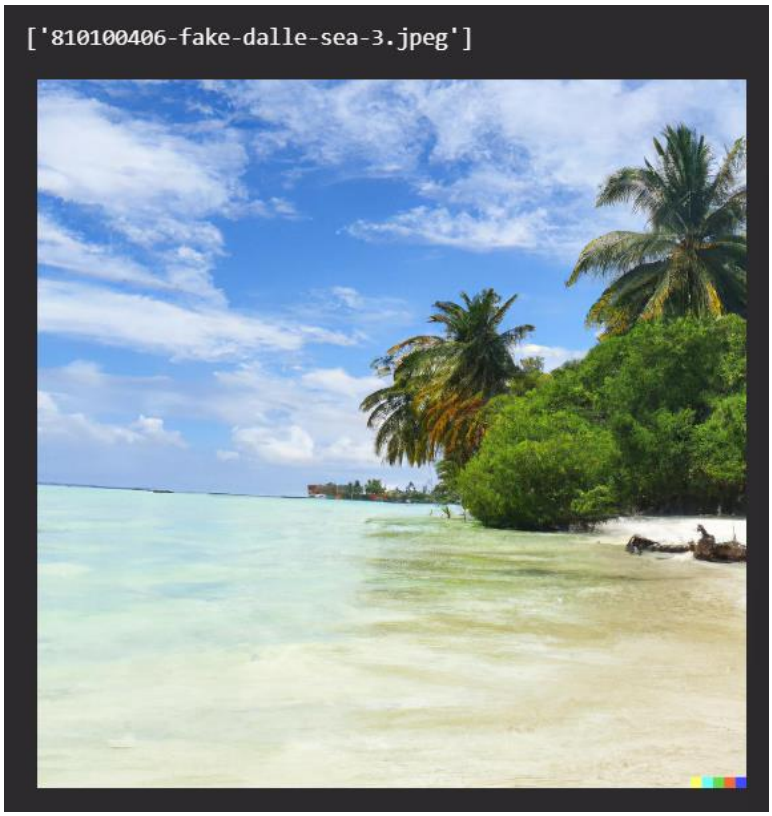


Figure14 fake but detected real

Figure15 **fake but detected real**

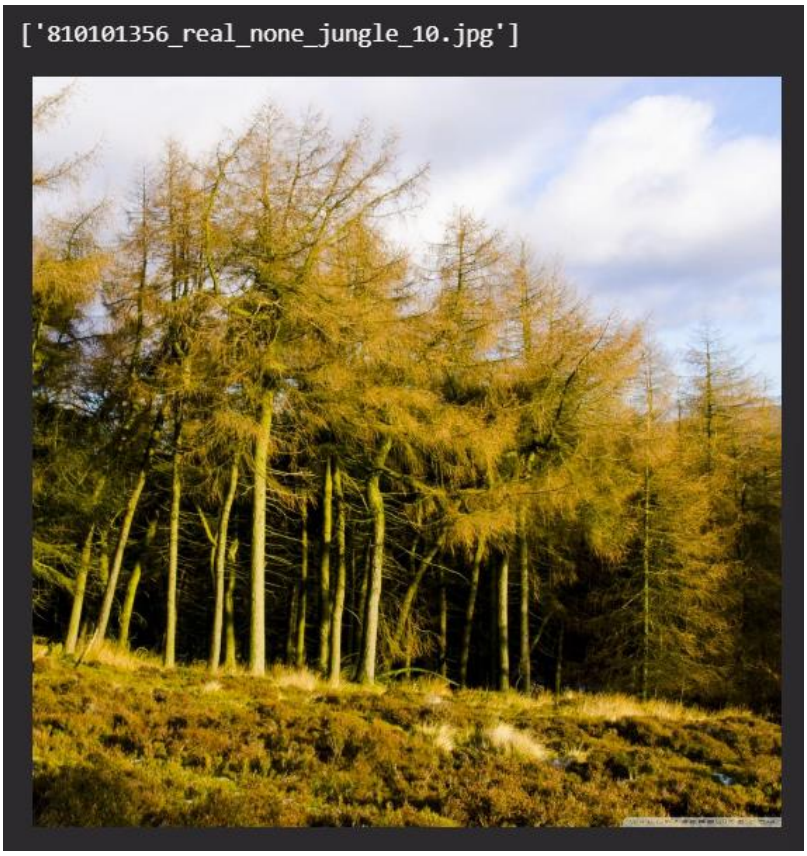


Figure16 **real but detected fake**

یکی از فیچر هایی که میتوان در نظر گرفت واتر مارک است که اگر در داده های دستیار آموزشی این فیچر در نظر گرفته می شد ، , Figure14دیگر اشتباه طبقه بندی نمیشدند .

طبقه بندی با استفاده از LDA

```
Confusion Matrix:
[[518   1]
 [  0 507]]

Accuracy: 0.9990253411306043
Error: 0.0009746588693957114
Precision: 0.9980314960629921
Recall: 1.0
F1-Score: 0.9990147783251232
```

Figure 17 ارزیابی مدل **LDA**

کاهش بعد به روش PCA و سپس طبقه بندی به روش SVM

```
Confusion Matrix:
[[517   2]
 [  0 507]]

Accuracy: 0.9980506822612085
Error: 0.001949317738791423
Precision: 0.9960707269155207
Recall: 1.0
F1-Score: 0.9980314960629921
```

Figure 18 ارزیابی مدل **SVM** پس از کاهش بعد

مقایسه طبقه بند ها : طبق خروجی های بالا ، طبقه بند LDA بهترین طبقه بند (با دقت بیشتر) است.

یعنی با استفاده از کاهش بعد LDA میتوانیم داده هارا با دقت بسیار بالایی تفکیک کنیم .

سوال چهارم: خوشه بندی

خوشه بندی با روش K-means

توضیح درباره ی معیار silhouette score برای ارزیابی خوشه :

silhouette معیاری است که برای ارزیابی کیفیت نتایج خوشه‌بندی استفاده می‌شود. فشردگی و تفکیک خوشه ها را در یک مجموعه داده اندازه گیری می کند. امتیاز silhouette از 1- تا 1 متغیر است، جایی که مقدار بالاتر، نشان دهنده نتایج خوشه بندی بهتر است.

خوشه بندی داده هایی که خودمان استخراج کرده ایم :

```
[1 1 1 ... 0 1 1]
----------------------------
Samples in Cluster 0: [0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
number of fake images =  475
number of real images =  729
----------------------------
----------------------------
Samples in Cluster 1: [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1,
number of fake images =  1216
number of real images =  974
----------------------------
Cluster 0: WCSS = 49162.969846975015
Cluster 1: WCSS = 92913.12117208062
silhouette_avg =  0.21588253269737606
```

Figure19 k=2

```
[2 2 2 ... 2 1 1]
----------------------------
Samples in Cluster 0: [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
number of fake images =  196
number of real images =  452
----------------------------
----------------------------
Samples in Cluster 1: [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
number of fake images =  510
number of real images =  421
----------------------------
----------------------------
Samples in Cluster 2: [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
number of fake images =  985
number of real images =  830
----------------------------
Cluster 0: WCSS = 27325.51478355835
Cluster 1: WCSS = 49812.71529078123
Cluster 2: WCSS = 48480.67064128199
silhouette_avg =  0.1704448608217166
```

Figure20 k=3

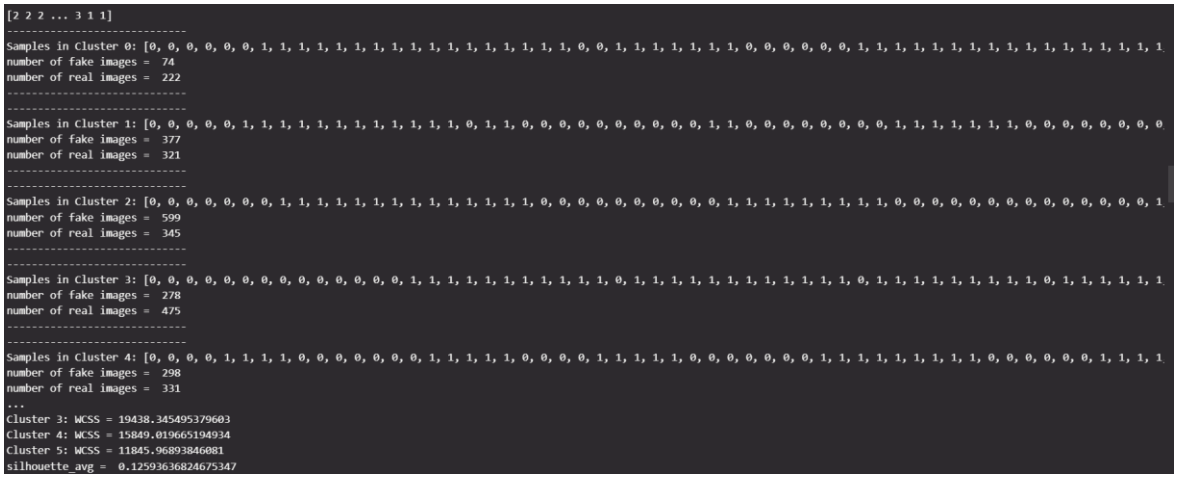


Figure21 **k=6**

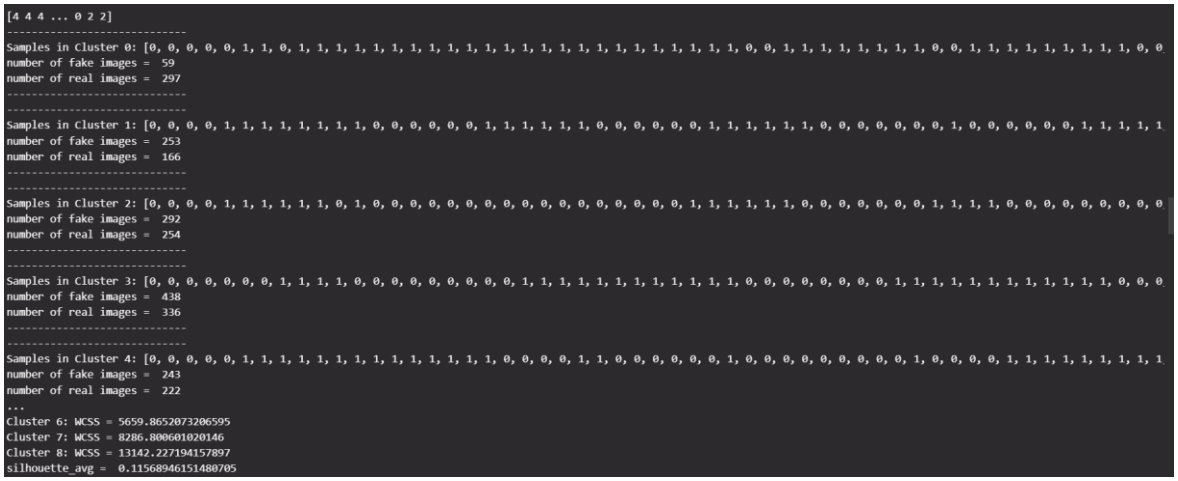


Figure22 **k=9**

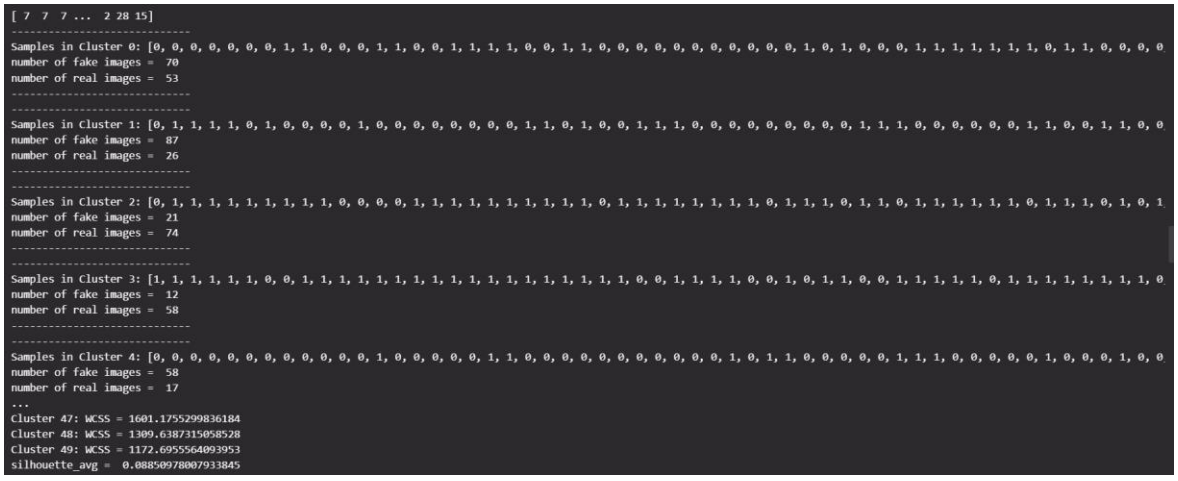


Figure23 **k=50**

همانطور که از خروجی کد بالا مشخص است ، با افزایش تعداد خوشه ها (k) ، معیار WCSS که نشاندهنده ی پراکنده بودن داده ها و فاصله داشتن داده ها از همدیگر در هر خوشه است ، کاهش می

یابد این بدین معناست که هر چقدر تعداد خوشه ها بیشتر باشد ، داده ها به صورت فشرده تر ، خوشه بندی و دسته بندی می شوند .

خوشه بندی با روش GMM

خوشه بندی داده هایی که خودمان استخراج کرده ایم :

```
gmm labels are :  [1 1 1 ... 0 1 1]

----------------------------
Samples in Cluster 0: [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1,
number of fake images =  828
number of real images =  1043
----------------------------
----------------------------
Samples in Cluster 1: [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1,
number of fake images =  863
number of real images =  660
----------------------------
silhouette_avg =  0.16124251400331208
```

Figure24 Number of component = 2

```
gmm labels are :  [1 1 1 ... 1 0 1]

----------------------------
Samples in Cluster 0: [0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 0,
number of fake images =  434
number of real images =  202
----------------------------
----------------------------
Samples in Cluster 1: [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1,
number of fake images =  864
number of real images =  817
----------------------------
----------------------------
Samples in Cluster 2: [0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
number of fake images =  393
number of real images =  684
----------------------------
silhouette_avg =  0.14053571302522544
```

Figure25 Number of component = 3

```
gmm labels are :  [5 5 5 ... 1 4 5]

----------------------------
Samples in Cluster 0: [0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1,
number of fake images =  285
number of real images =  296
----------------------------
----------------------------
Samples in Cluster 1: [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
number of fake images =  236
number of real images =  416
----------------------------
----------------------------
Samples in Cluster 2: [0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
number of fake images =  100
number of real images =  254
----------------------------
----------------------------
Samples in Cluster 3: [0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0,
number of fake images =  85
number of real images =  10
----------------------------
----------------------------
Samples in Cluster 4: [0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 1, 1, 1, 0, 0,
number of fake images =  299
...
number of fake images =  686
number of real images =  545
----------------------------
silhouette_avg =  0.10454270576321623
```

Figure26 **Number of component = 6**

```
gmm labels are :  [1 1 1 ... 3 2 1]

----------------------------
Samples in Cluster 0: [0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0,
number of fake images =  344
number of real images =  253
----------------------------
----------------------------
Samples in Cluster 1: [0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
number of fake images =  425
number of real images =  354
----------------------------
----------------------------
Samples in Cluster 2: [0, 0, 0, 1, 1, 1, 0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 1,
number of fake images =  242
number of real images =  108
----------------------------
----------------------------
Samples in Cluster 3: [0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1,
number of fake images =  280
number of real images =  337
----------------------------
----------------------------
Samples in Cluster 4: [0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
number of fake images =  80
...
number of fake images =  187
number of real images =  125
----------------------------
silhouette_avg =  0.06711927823464806
```

Figure27 **Number of component = 9**

```
gmm labels are :  [43 43 43 ... 23 46  9]

----------------------------
Samples in Cluster 0: [1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 1,
number of fake images =  31
number of real images =  23
----------------------------
----------------------------
Samples in Cluster 1: [0, 1, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0,
number of fake images =  78
number of real images =  23
----------------------------
----------------------------
Samples in Cluster 2: [0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 1,
number of fake images =  149
number of real images =  60
----------------------------
----------------------------
Samples in Cluster 3: [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0,
number of fake images =  56
number of real images =  10
----------------------------
----------------------------
Samples in Cluster 4: [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
number of fake images =  4
...
number of fake images =  0
number of real images =  3
----------------------------
silhouette_avg =  0.07811695649736337
```

Figure 28 Number of component = 50

خوشه بندی با روش K-means(TA data)

خوشه بندی داده هایی که دستیار آموزشی استخراج کرده است :

```
kmean labels are :  [1 1 0 ... 1 0 0]
----------------------------
Samples in Cluster 0: [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
number of fake images =  9
number of real images =  1696
----------------------------
----------------------------
Samples in Cluster 1: [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
number of fake images =  1700
number of real images =  12
----------------------------
Cluster 0: WCSS = 1630947.3128300216
Cluster 1: WCSS = 1355618.4145731388
silhouette_avg =  0.2963976104593302
```

Figure 29 k = 2

```
kmean labels are :  [1 1 0 ... 1 0 2]
----------------------------
Samples in Cluster 0: [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
number of fake images =  0
number of real images =  702
----------------------------
----------------------------
Samples in Cluster 1: [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
number of fake images =  1440
number of real images =  0
----------------------------
----------------------------
Samples in Cluster 2: [1, 1, 1, 0, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
number of fake images =  269
number of real images =  1006
----------------------------
Cluster 0: WCSS = 779789.2587493927
Cluster 1: WCSS = 1117745.4785661008
Cluster 2: WCSS = 630426.2982630836
silhouette_avg =  0.2308347918417357
```

Figure 30 k = 3

```
kmean labels are :  [0 0 1 ... 0 4 1]
----------------------------
Samples in Cluster 0: [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
number of fake images =  459
number of real images =  0
----------------------------
----------------------------
Samples in Cluster 1: [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
number of fake images =  2
number of real images =  1144
----------------------------
----------------------------
Samples in Cluster 2: [0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0,
number of fake images =  525
number of real images =  88
----------------------------
----------------------------
Samples in Cluster 3: [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
number of fake images =  127
number of real images =  0
----------------------------
----------------------------
Samples in Cluster 4: [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
number of fake images =  0
number of real images =  476
...
Cluster 3: WCSS = 131353.48974934418
Cluster 4: WCSS = 557695.4364806942
Cluster 5: WCSS = 261139.78839870263
silhouette_avg =  0.17119420182888595
```

Figure 31 k = 6

```
kmean labels are :  [7 5 6 ... 7 6 1]
----------------------------
Samples in Cluster 0: [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
number of fake images =  0
number of real images =  262
----------------------------
----------------------------
Samples in Cluster 1: [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
number of fake images =  37
number of real images =  817
----------------------------
----------------------------
Samples in Cluster 2: [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
number of fake images =  409
number of real images =  0
----------------------------
----------------------------
Samples in Cluster 3: [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
number of fake images =  81
number of real images =  0
----------------------------
----------------------------
Samples in Cluster 4: [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
number of fake images =  0
number of real images =  147
...
Cluster 6: WCSS = 267229.9560146999
Cluster 7: WCSS = 203780.04754441534
Cluster 8: WCSS = 49561.89961475571
silhouette_avg =  0.18607277783694567
```

Figure 32 k = 9

```
kmean labels are :  [16 46 22 ... 20  2 27]
----------------------------
Samples in Cluster 0: [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
number of fake images =  0
number of real images =  145
----------------------------
----------------------------
Samples in Cluster 1: [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
number of fake images =  70
number of real images =  0
----------------------------
----------------------------
Samples in Cluster 2: [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
number of fake images =  0
number of real images =  85
----------------------------
----------------------------
Samples in Cluster 3: [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
number of fake images =  49
number of real images =  0
----------------------------
----------------------------
Samples in Cluster 4: [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
number of fake images =  51
number of real images =  0
...
Cluster 47: WCSS = 26287.218520406073
Cluster 48: WCSS = 9797.056394326231
Cluster 49: WCSS = 5272.478334728319
silhouette_avg =  0.11017937473933023
```

Figure 33 k = 50

با توجه به نتایج بالا در میابیم که خوشه بندی روی داده هایی که دستیار آموزشی در اختیار ما قرار داده است ، به خوبی و با دقت بالایی انجام می شود برای مثال در بعضی از خوشه ها فقط داده های یک کلاس از دو کلاس real , fake قرار میگیرد که این نشاندهنده ی خوشه بندی مناسب است .

خوشه بندی با روش (TA)GMM

```
gmm labels are :  [0 0 1 ... 0 1 1]

----------------------------
Samples in Cluster 0: [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
number of fake images =  1700
number of real images =  12
----------------------------
----------------------------
Samples in Cluster 1: [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
number of fake images =  9
number of real images =  1696
----------------------------
silhouette_avg =  0.2963976104593302
```

Figure34 Number of component = 2

```
gmm labels are :  [1 1 0 ... 1 0 2]

----------------------------
Samples in Cluster 0: [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
number of fake images =  0
number of real images =  734
----------------------------
----------------------------
Samples in Cluster 1: [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
number of fake images =  1417
number of real images =  0
----------------------------
----------------------------
Samples in Cluster 2: [1, 1, 1, 0, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
number of fake images =  292
number of real images =  974
----------------------------
silhouette_avg =  0.2277237767092863
```

Figure 35 Number of component = 3

```
gmm labels are :  [1 4 2 ... 1 3 0]

----------------------------
Samples in Cluster 0: [1, 1, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0,
number of fake images =  143
number of real images =  687
----------------------------
----------------------------
Samples in Cluster 1: [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
number of fake images =  966
number of real images =  0
----------------------------
----------------------------
Samples in Cluster 2: [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
number of fake images =  0
number of real images =  734
----------------------------
----------------------------
Samples in Cluster 3: [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
number of fake images =  0
number of real images =  287
----------------------------
----------------------------
Samples in Cluster 4: [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
number of fake images =  336
...
number of fake images =  264
number of real images =  0
----------------------------
silhouette_avg =  0.19352572102071752
```

Figure 36 Number of component = 6

```
gmm labels are :  [1 4 5 ... 1 5 0]

----------------------------
Samples in Cluster 0: [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
number of fake images =  49
number of real images =  791
----------------------------
----------------------------
Samples in Cluster 1: [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
number of fake images =  702
number of real images =  2
----------------------------
----------------------------
Samples in Cluster 2: [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
number of fake images =  0
number of real images =  176
----------------------------
----------------------------
Samples in Cluster 3: [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
number of fake images =  264
number of real images =  0
----------------------------
----------------------------
Samples in Cluster 4: [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
number of fake images =  285
...
number of fake images =  0
number of real images =  224
----------------------------
silhouette_avg =  0.18526180351059138
```

Figure 37 Number of component = 9

```
gmm labels are :  [32 10 49 ... 44  2 36]

----------------------------
Samples in Cluster 0: [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
number of fake images =  0
number of real images =  115
----------------------------
----------------------------
Samples in Cluster 1: [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
number of fake images =  91
number of real images =  0
----------------------------
----------------------------
Samples in Cluster 2: [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
number of fake images =  0
number of real images =  84
----------------------------
----------------------------
Samples in Cluster 3: [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
number of fake images =  69
number of real images =  0
----------------------------
----------------------------
Samples in Cluster 4: [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
number of fake images =  20
...
number of fake images =  0
number of real images =  85
----------------------------
silhouette_avg =  0.1072568525294598
```

Figure 38 Number of component = 50

خوشه بندی با روش K-means و بررسی چگونگی تفکیک کوه و دریا و جنگل:

- K=2:

```
----------------------------
Samples in Cluster 0:
number of sea images =  587
number of jungle images =  556
number of mountain images =  562
----------------------------
Samples in Cluster 1:
number of sea images =  552
number of jungle images =  582
number of mountain images =  578
silhouette_avg =  0.2963976104593302
```

- K=3:

```
----------------------------
Samples in Cluster 0:
number of sea images =  432
number of jungle images =  414
number of mountain images =  420
----------------------------
Samples in Cluster 1:
number of sea images =  243
number of jungle images =  254
number of mountain images =  237
----------------------------
Samples in Cluster 2:
number of sea images =  464
number of jungle images =  470
number of mountain images =  483
silhouette_avg =  0.2277237767092863
```

- K=6:

```
Samples in Cluster 0:
number of sea images =  89
number of jungle images =  86
number of mountain images =  89
----------------------------
Samples in Cluster 1:
number of sea images =  257
number of jungle images =  223
number of mountain images =  254
----------------------------
Samples in Cluster 2:
number of sea images =  111
number of jungle images =  109
number of mountain images =  116
----------------------------
Samples in Cluster 3:
number of sea images =  100
number of jungle images =  107
number of mountain images =  80
----------------------------
Samples in Cluster 4:
number of sea images =  274
number of jungle images =  289
number of mountain images =  267
----------------------------
Samples in Cluster 5:
number of sea images =  308
number of jungle images =  324
number of mountain images =  334
silhouette avg =  0.19352572102071752
```

- K=9:

```
----------------------------
Samples in Cluster 0:
number of sea images =  169
number of jungle images =  150
number of mountain images =  171
----------------------------
Samples in Cluster 1:
number of sea images =  105
number of jungle images =  131
number of mountain images =  145
----------------------------
Samples in Cluster 2:
number of sea images =  196
number of jungle images =  213
number of mountain images =  182
----------------------------
Samples in Cluster 3:
number of sea images =  106
number of jungle images =  107
number of mountain images =  121
----------------------------
Samples in Cluster 4:
number of sea images =  300
number of jungle images =  277
number of mountain images =  274
----------------------------
Samples in Cluster 5:
number of sea images =  51
number of jungle images =  54
number of mountain images =  47
----------------------------
Samples in Cluster 6:
number of sea images =  41
number of jungle images =  33
number of mountain images =  42
----------------------------
Samples in Cluster 7:
number of sea images =  90
number of jungle images =  85
number of mountain images =  83
----------------------------
Samples in Cluster 8:
number of sea images =  81
number of jungle images =  88
number of mountain images =  75
silhouette avg =  0.1830893395296088
```

- K=50:

چند تا آخرین کلاسترها:

```
----------------------------
Samples in Cluster 44:
number of sea images =  23
number of jungle images =  19
number of mountain images =  24
----------------------------
Samples in Cluster 45:
number of sea images =  5
number of jungle images =  7
number of mountain images =  6
----------------------------
Samples in Cluster 46:
number of sea images =  23
number of jungle images =  19
number of mountain images =  32
----------------------------
Samples in Cluster 47:
number of sea images =  25
number of jungle images =  28
number of mountain images =  19
----------------------------
Samples in Cluster 48:
number of mountain images =  2
----------------------------
Samples in Cluster 49:
number of sea images =  12
number of jungle images =  13
number of mountain images =  12
silhouette_avg =  0.10847827317967765
```

توزیع متوازنی از تصاویر کوه و دریا و جنگل در هر کلاستر وجود دارد چون فیچرهای ما بر اساس fake و real بودن عکس ها بود.

توضیح درباره ی خوشه بندی ها : میدانیم دو کلاس واقعی و ساختگی داریم . بنابراین باید بتوانیم با دو خوشه این داده ها را دسته بندی کنیم . حال اگر تعداد کلاستر ها را بیشتر کنیم ، این دسته بندی با جزئیات بیشتری انجام می شود یعنی مثلا در خوشه واقعی ، میتوانیم دریا یا کوه یا جنگل را نیز دسته بندی کنیم . بنابراین بیشتر کردن کلاستر (خوشه) این فایده را دارد .