(به نام خداوند مهربان)

پروژه دوره Data Analysis مجتمع فنی

خوشه بندی داده های کالج های ایالات متحده

U.S. News and World Report's College Data

(Clustering)

استاد: مهندس جناب آقای ولدبیگی

نام تحلیلگر: مهدی عماری

دیتاست : https://www.kaggle.com/datasets/flyingwombat/us-news-and-world-reports-college-data

توضیحات درباره دیتاست :

U.S. News and World Report's College Data یک مجموعه اطلاعات است که توسط مجله U.S. News and World Report جمع آوری شده و ارائه شده است. این دیتاست مربوط به اطلاعات مختلف مربوط به دانشگاه‌ها است. این اطلاعات می‌توانند شامل مواردی مانند تعداد دانشجویان پذیرفته شده، تعداد دانشجویانی که واجد شرایط ورود از ده درصد برتر هستند، تعداد دانشجویانی که از ۲۵ درصد برتر هستند، تعداد دانشجویانی که پذیرفته شده‌اند و وارد دانشگاه شده‌اند، تعداد دانشجویان فارغ‌التحصیل، هزینه‌های ایاب و ذهاب، هزینه‌های اقامت، هزینه‌های کتاب، هزینه‌های شخصی، درصد اساتید با مدرک دکتری، درصد اساتیدی که مدرک دکتری دارند، نسبت دانشجو به اساتید، درصد فارغ‌التحصیلانی که از دانشگاه حمایت می‌کنند، هزینه‌های انتقال و نرخ فارغ‌التحصیلی باشد

این داده‌ها به داوطلبان و دانشجویان کمک می‌کند تا تصمیمات درستی در مورد انتخاب دانشگاه برای تحصیلات خود بگیرند همچنین این اطلاعات می‌توانند برای مدل‌سازی و پیش‌بینی‌های مختلف مربوط به دانشگاه‌ها مورد استفاده قرار بگیرند، از جمله پیش‌بینی نرخ فارغ‌التحصیلی، شهرت دانشگاه، و موارد دیگر.

این دیتاست به نوع داده‌های عددی (numeric data) و داده‌های دسته‌ای (categorical data) تقسیم می‌شود. به عنوان مثال، هزینه‌های انتقال و نرخ فارغ‌التحصیلی به عنوان داده‌های عددی در نظر گرفته می‌شوند، در حالی که مواردی مانند نام دانشگاه و دانشگاه (خصوصی یا دولتی) به عنوان داده‌های دسته‌ای محسوب می‌شوند.

به عنوان مثال، برای داده‌های عددی می‌توان از رگرسیون خطی و شبکه‌های عصبی استفاده کرد، در حالی که برای داده‌های دسته‌ای می‌توان از مدل‌های دسته‌بندی و خوشه‌بندی استفاده نمود.

برای دیتاستی که شامل اطلاعات عددی و دسته‌ای است، می‌توان الگوریتم‌های متنوعی را پیاده‌سازی کرد

۱) Regression Algorithms: برای پیش‌بینی و تحلیل رابطه‌های میان ویژگی‌های مختلف مانند هزینه‌های تحصیلی و ویژگی‌های دانشگاهی. می‌توان از الگوریتم‌های رگرسیون مانند Linear Regression یا Decision Tree Regression و.... استفاده کرد. ✖

۲) Classification Algorithms: اگر می‌خواهید یک مدل بسازید که بتواند داده‌ها را به چند دسته مختلف تقسیم کند (مانند رتبه‌بندی دانشگاه‌ها بر اساس نرخ فارغ‌التحصیلی)، می‌توانید از مدل‌های کلاسیفیکیشن استفاده کنید مدل‌های دسته‌بندی برای تحلیل و پیش‌بینی وضعیت دانشگاه‌ها، مانند دانشگاه‌های خصوصی یا دولتی بکار میرود ✖

۳) Clustering Algorithms: اگر می‌خواهید داده‌ها را به گروه‌های مشخصی تقسیم کنید بدون داشتن برچسب‌های قبلی (مانند گروه‌بندی دانشگاه‌ها بر اساس ویژگی‌هایشان)، می‌توانید از مدل‌های کلاسترینگ مانند K-means استفاده کنید ✓

تحلیل روی کد :

۱) وارد کردن دیتاست (دیتاست کالج)

۲) Preprocessing یا پیش پردازش دیتا:

یک نکته که اینجا هستش اینه که باید توجه داشته باشید که یک متغیر دسته بندی در داده های ما وجود دارد (خصوصی) متغیرهای طبقه بندی برای خوشه بندی مشکل دارند. شما نمی توانید یک متغیر طبقه بندی شده را خوشه بندی کنید، بنابراین باید نوعی نقشه برداری برای آن انجام دهید. این می‌تواند برای داده‌های ترتیبی شهودی باشد، اما برای متغیرهای طبقه‌بندی غیرترتیبی، تخصیص مقادیر عددی می‌تواند روی خوشه‌ها به‌گونه‌ای تأثیر بگذارد که در مورد داده‌های اساسی معنادار نیست. Private یک متغیر باینری است، بله یا خیر، اما نگاشت ۰ یا ۱ تاثیر بزرگی بر خوشه بندی خواهد داشت، زیرا هر نقطه تماماً در حداقل یا حداکثر این متغیر خواهد بود در حالی که سایر متغیرها پیوسته خواهند بود. در حال حاضر، ما این متغیر را نادیده می گیریم.

```
features = df.drop(['Private', 'Unnamed: 0'],axis=1)
```

ستون طبقه بندی و نام کالج را حذف کردیم

```
features['Acceptperc'] = features['Accept'] / features['Apps']
features['Enrollperc'] = features['Enroll'] / features['Accept']
```

این دو خط کد که من ارائه نوشتم دو ستون جدید در یک دیتافریم features ایجاد می‌کنند که شامل داده‌های مربوط به پذیرش دانشگاه‌ها است:

Acceptpercاین ستون نسبت تعداد پذیرش‌ها (Accept) به تعداد درخواست‌های ارسال شده (Apps)را نشان می‌دهد و به عنوان درصد پذیرش تعبیر می‌شود.

۲ :Enrollperc .این ستون نسبت تعداد دانشجویانی که ثبت نام کرده‌اند (Enroll) به تعداد دانشجویانی که پذیرش شده‌اند (Accept) را نشان می‌دهد و به عنوان درصد ثبت نام از میان پذیرفته‌شدگان تعبیر می‌شود.

این محاسبات به شما کمک می‌کنند تا درصدهایی را برای فهم بهتر عملکرد و جذابیت دانشگاه‌ها از منظر داوطلبان و پذیرفته‌شدگان به دست آورید این می‌تواند به تحلیل‌های بعدی در مورد کیفیت دانشگاه‌ها یا اثربخشی استراتژی‌های جذب دانشجو آن‌ها کمک کند.

توجه داشته باشید که دسته بندی های مختلف محدوده های متفاوتی دارند. اگر آنها را عادی نکنیم، ستون‌هایی با دامنه وسیع‌تر سهم نامتناسبی در جداسازی خوشه‌ها خواهند داشت.

برای همین نرمال‌سازی میکینم.....

نرمال‌سازی (Normalization) در داده‌کاوی و یادگیری ماشین، فرآیندی است که طی آن مقادیر ویژگی‌ها (features) در یک دامنه‌ی مشخص استانداردسازی می‌شوند. هدف از نرمال‌سازی، اصلاح دامنه مقادیر ویژگی‌ها به گونه‌ای است که مقیاس‌های مختلف را به یک مقیاس مشترک بیاورد، بدین صورت که الگوریتم‌های یادگیری ماشین بتوانند بهتر کار کنند و مقایسه ویژگی‌ها راحت‌تر شود. نرمال‌سازی می‌تواند تاثیر یکسانی میان ویژگی‌هایی با مقیاس‌های

متفاوت ایجاد کند و از این جهت می‌تواند به بهبود کارایی ویژگی‌ها در مدل‌های یادگیری ماشین کمک کند.

چند روش معمول نرمال‌سازی عبارت‌اند از:

۱. **نرمال‌سازی مین‌مکس :(Min-Max Normalization)** این روش هر مقدار را بین ۰ و ۱ تنظیم می‌کند.
۲. **نرمالسازی-Z اسکور :(Z-Score Normalization)** این روش میانگین را می‌گیرد و توسط انحراف معیار تقسیم می‌کند تا ویژگی‌ها را به میانگین صفر و انحراف معیار یک تبدیل کند
۳. **نرمال‌سازی مقیاس‌بندی :(Scaling)** تغییر مقیاس مقادیر به صورتی که همه ویژگی‌ها در یک محدوده مشخص قرار بگیرند، معمولاً بین صفر و یک یا منهای یک تا یک.
۴. نرمال‌سازی L1 و L2 :نرمال‌سازی (L1) و (L2) دو نوع از نرمال‌سازی هستند که اغلب در پیش‌پردازش داده‌ها برای الگوریتم‌های یادگیری ماشین استفاده می‌شوند. این دو روش به طور خاص برای تغییر مقیاس و نوع توزیع داده‌ها طراحی شده‌اند تا تاثیر ناهمواری‌ها روی الگوریتم‌های یادگیری کاهش یابد.

L1: **نرمال‌سازی (L1) (یا نرمال‌سازی منهتن، یا نرم برداری- نرم۱):** این نرمال‌سازی اغلب به عنوان نرمال‌سازی منهتن شناخته می‌شود چرا که براساس مسافت منهتن، که مجموع مطلق اختلافات در همه بعدهاست کار می‌کند در این روش مقادیر یک ویژگی یا سطر به گونه‌ای تغییر مقیاس می‌یابند که مجموع مقادیر مطلق آن‌ها برابر با ۱ شود. این کار اغلب برای تهیه داده‌ها قبل از استفاده از الگوریتم‌هایی که به مقیاس حساس نیستند انجام می‌شود.

L2: **نرمال‌سازی(L2) (یا نرمال‌سازی اقلیدسی، یا نرم برداری -نرم۲):**
نرمال‌سازی (L2) بر مبنای فاصله اقلیدسی انجام می‌شود، که مربع جذر مجموع مربعات اختلاف‌ها در همه بعدهاست. در این روش، مقادیر یک ویژگی یا سطر به نحوی تغییر مقیاس می‌یابند که مجموع مربع مقادیر آن‌ها (همچنین شناخته شده به عنوان فاصله

اقلیدسی) برابر با ۱ شود. این اغلب به عنوان یک روش برای کاهش تاثیرoutlier ها در داده‌ها به کار برده می‌شود، چون مربعات مقادیر بزرگتر وزن بیشتری پیدا می‌کنند و در نتیجه تاثیرشان کمتر می‌شود.

هر دو روش کاربردهای خاص خود را دارند و بسته به مسئله و مدل یادگیری مورد استفاده، می‌توان یکی را انتخاب کرد.

انتخاب روش نرمال‌سازی بستگی به داده‌ها و الگوریتم مورد استفاده در مدل یادگیری ماشین دارد. برخی مدل‌ها، مانند شبکه‌های عصبی، کارایی بسیار بهتری دارند وقتی که ورودی‌ها نرمال‌سازی شده‌اند، در حالی که برای دیگر مدل‌ها، مانند درخت‌های تصمیم، نرمال‌سازی الزامی نیست.

```
scaler = preprocessing.MinMaxScaler()
features_normal = scaler.fit_transform(features)
```

من برای نرمال‌سازی ویژگی‌های (features) که دیتافریمم است با روش Min-Max Scaling این کد من یک نرمال‌ساز Min-Max جدید ایجاد می‌کند و سپس این نرمال‌ساز را با استفاده از داده‌های موجود در دیتافریم features آموزش می‌دهد و نهایتاً این نرمال‌ساز را برای تغییر مقیاس ویژگی‌های داده‌ها استفاده می‌کند به طوری که همه آن‌ها در محدوده [۰, ۱] قرار بگیرند.

بگذارید تک تک این مراحل را بررسی کنیم:

۱: scaler = preprocessing.MinMaxScaler() شیء scaler که یک نمونه از کلاس MinMaxScaler از ماژول preprocessing کتابخانهscikit-learn ما است ساخته می‌شود. این شیء برای نرمال‌سازی داده‌ها بین صفر و یک پیکربندی شده است.

۲: features_normal = scaler.fit_transform(features) .متد fit_transform ابتدا مدل نرمال‌سازی را با features آموزش می‌دهد (یعنی محدوده‌های داده را برای هر ستون محاسبه می‌کند) و سپس داده‌ها را نرمال‌کرده و به صورت نرمال‌شده خروجی می‌دهد.

پس از انجام این مراحل features_normal یک آرایه‌ی نامپای (NumPy array) خواهد بود که هر ویژگی در آن بین ۰ و ۱ نرمال‌شده است. این عمل باعث سهولت در یادگیری و بهبود عملکرد برخی الگوریتم‌های ماشین یادگیری می‌شود، زیرا تمام ویژگی‌ها دارای وزن‌های مشابه‌اند و هیچ یک به صورت ناعادلانه‌ای بر تابع هزینه تأثیر نخواهد گذاشت.

---

pd.DataFrame(features_normal).describe()

[30] ✓ 0.1s Python

	0	1	2	3	4	5	6	7	8	9
count	777.000000	777.000000	777.000000	777.000000	777.000000	777.000000	777.000000	777.000000	777.000000	777.000000
mean	0.060830	0.074141	0.117189	0.279564	0.514249	0.113030	0.039125	0.418423	0.406294	0.202041
std	0.080607	0.093347	0.146166	0.185688	0.217635	0.153962	0.069724	0.207800	0.172871	0.073576
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	0.014475	0.020260	0.032563	0.147368	0.351648	0.027076	0.004305	0.257231	0.286412	0.166667
50%	0.030763	0.039531	0.062765	0.231579	0.494505	0.049771	0.016121	0.395145	0.381463	0.180036
75%	0.073793	0.089573	0.136385	0.357895	0.659341	0.122715	0.044241	0.546746	0.515448	0.224599
max	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000

وقتی از تابع describe() روی دیتافریمم تشکیل شده از features_normal که نتیجه‌ی نرمال‌سازی ویژگی‌ها با استفاده از Min-Max Scaling است استفاده کردم در حال درخواست یک خلاصه آماری از داده‌های نرمال شده هستم ، آمار توصیفی از شامل تعداد، میانگین، انحراف معیار، حداقل مقدار، ۲۵ درصد پایین، میانه (۵۰ درصد)، ۷۵ درصد بالا و حداکثر مقدار را برای هر ستون در دیتافریم ارائه کرد.

با توجه به اینکه شما ویژگی‌هایتان را با Min-Max Scaling نرمال کرده‌اید، در اینجا چند نتیجه‌گیری است که می‌توان انتظار داشت:

count -نشان دهنده تعداد داده‌ها در هر ستون است که باید برای همه ویژگی‌ها یکسان باشد مگر اینکه داده گمشده وجود داشته باشد.

mean -میانگین مقادیر برای هر ستون بعد از نرمال‌سازی، که نزدیک به ۰.۵ خواهد بود اگر داده‌ها به‌طور یکنواخت در محدوده [۰, ۱] پخش شده باشند.

std -انحراف معیار مقادیر در هر ستون را نشان می‌دهد که اطلاعاتی درباره پراکندگی داده‌ها می‌دهد.

min -حداقل مقدار برای هر ستون که باید صفر باشد (مینیمم مقدار بعد از نرمال‌سازیMin-Max)

25% -نشان‌دهنده مقداری است که ۲۵ درصد داده‌ها کمتر یا برابر با آن هستند.

50% -نشان‌دهنده مقدار میانه است، که در نرمال‌سازی Min-Max اغلب بسیار نزدیک به ۰.۵ خواهد بود.

75% -نشان‌دهنده مقداری است که ۷۵ درصد داده‌ها زیر یا برابر با آن هستند.

max -حداکثر مقدار برای هر ویژگی که باید یک باشد(ماکسیمم مقدار بعد از نرمال‌سازی.Min-Max)

این معیارها می‌توانند در شناسایی خصوصیات کلیدی دیتاست ما پس از نرمال‌سازی مانند انتشار ویژگی‌ها و شناسایی خطاهای احتمالی در پیش‌پردازش کمک کننده باشند.

---

۳)اطلاعات در مورد داده ها (EDA)

ما برای انجام Exploratory Data Analysis (EDA) بر روی دیتاستمان، اقداماتی انجام دادیم از جمله:

۱. **تجزیه و تحلیل توصیفی**: بررسی آمارهای توصیفی مانند میانگین، میانه، واریانس و کوچکترین/بزرگترین مقادیر هر ویژگی(این کار را مرحله قبل انجام دادم)

---

۲. **بررسی روابط:** بررسی روابط و ارتباطات بین ویژگی‌ها، به دنبال ویژگی‌هایی که با یکدیگر همبستگی دارند یا تاثیر یکدیگر را می‌توانند بررسی کنید.

```
instution_type =
colleges.groupby('Private')[['Apps','Accept','Enroll','F.Undergrad','P.Und
ergrad']].sum().reset_index()

instution_type['Expend'] =
colleges.groupby('Private').Expend.mean().reset_index()['Expend']

st = instution_type['Enroll']

instution_type['Student_percentage'] =
[st[0]/(st[0]+st[1]),st[1]/(st[0]+st[1])]

instution_type
```

این کد روی دیتاست اصلی انجام شده یعنی قبل از پیش پردازش برای بدست اوردن یک اماری

کدی که اینجا ارائه دادم یک دیتافریم جدید به نام instution_type ایجاد می‌کند که آمار مربوط به تعداد درخواست‌ها پذیرفته‌شدگان تعداد دانشجویان ثبت‌نام کرده دانشجویان کارشناسی پیوسته و کارشناسی ناپیوسته را برای دو نوع دانشگاه خصوصی (Private) و دولتی (غیرخصوصی) جمع‌آوری می‌کند سپس میانگین هزینه‌های صرف شده برای هر نوع مؤسسه آموزشی به هر رکورد اضافه می‌کند و در نهایت درصد دانشجویان ثبت‌نام کرده در هر نوع مؤسسه را محاسبه و به دیتافریم اضافه می‌کند.

این کد را به صورت زیر توضیح دادم:

۱ .ایجاد دیتافریم instution_type با گروه‌بندی داده‌هایDF (دیتافریم اصلی)بر اساس ستون Private(قبل ازپیش پردازش) و محاسبه جمع ستون‌های مشخص شده.

۲ .اضافه کردن ستون Expend با محاسبه میانگین ستون Expend برای هر گروه از دانشگاه‌های خصوصی و غیرخصوصی.

۳ .ذخیره ستون Enroll از دیتافریم instution_type در متغیرst

۴ .محاسبه و اضافه کردن ستون جدید به نام Student_percentage با استفاده از درصد تعداد ثبت‌نام‌کرده‌ها نسبت به کل تعداد ثبت‌نام‌کرده‌ها در هر دو نوع دانشگاه.

این کد ما باید پایانی با صدا زدن instution_type داشته باشد تا دیتافریم نهایی نمایش داده شود سپس دیتافریم `instution_type` را مشاهده کنید

خروجی ما دیتافریمی است که شامل داده‌های تجمیع‌شده و اطلاعات درصدی است که شرح داده شده‌اند

	Private	Apps	Accept	Enroll	F.Undergrad	P.Undergrad	Expend	Student_percentage
0	No	1214743	830889	347865	1817053	419376	7458.316038	0.573998
1	Yes	1117530	737722	258174	1057775	245191	10486.353982	0.426002

+ Code + Markdown

این دیتافریم ما جدولی است که داده‌های جمع‌آوری شده از دانشگاه‌های خصوصی و غیرخصوصی (دولتی) را نشان می‌دهد برای هر نوع از این دانشگاه‌ها شماره‌ای از کلیدهای آماری موجود است در ادامه تحلیل من از اطلاعات ارائه شده در هر ردیف را می‌بینید:

دانشگاه‌های غیرخصوصی (ردیف با شماره ۰):

Apps -(تعداد درخواست‌ها): ۱,۲۱۴,۷۴۳ درخواست برای پذیرش به دانشگاه‌های غیرخصوصی ارسال شده است

Accept -(تعداد پذیرش‌ها): از این تعداد ۸۳۰,۸۸۹ درخواست پذیرفته شده‌اند

Enroll -(تعداد ثبت‌نام‌ها): در نهایت ۳۴۷,۸۶۵ نفر برای شرکت در دوره‌های این دانشگاه‌ها ثبت‌نام کرده‌اند

F.Undergrad -(تعداد دانشجویان کارشناسی پیوسته): مجموع دانشجویان کارشناسی پیوسته در این دانشگاه‌ها ۱,۸۱۷,۰۵۳ نفر است

P.Undergrad -(تعداد دانشجویان کارشناسی ناپیوسته): در مقابل، ۴۱۹,۳۷۶ دانشجوی کارشناسی ناپیوسته در این دانشگاه‌ها تحصیل می‌کنند

Expend -(هزینه‌ها): میانگین هزینه صرف شده برای هر دانشجو در این نوع دانشگاه‌ها ۷,۴۵۸ دلار است

Student_percentage -(درصد دانشجویان): حدود ۵۷.۴٪ از کل دانشجویان ثبت‌نام کرده در هر دو نوع دانشگاه در دانشگاه‌های غیرخصوصی ثبت‌نام کرده‌اند

دانشگاه‌های خصوصی (ردیف با شماره ۱):

Apps -(تعداد درخواست‌ها): ۱,۱۱۷,۵۳۰ درخواست برای پذیرش به دانشگاه‌های خصوصی ارسال شده است

Accept -(تعداد پذیرش‌ها): از این تعداد، ۷۳۷,۷۲۲ درخواست پذیرفته شده‌اند

Enroll -(تعداد ثبت‌نام‌ها): در نهایت، ۲۵۸,۱۷۴ نفر برای شرکت در دوره‌های این دانشگاه‌ها ثبت‌نام کرده‌اند

F.Undergrad -(تعداد دانشجویان کارشناسی پیوسته): مجموع دانشجویان کارشناسی پیوسته در این دانشگاه‌ها ۱,۰۵۷,۷۷۵ نفر است

P.Undergrad -(تعداد دانشجویان کارشناسی ناپیوسته): در مقابل، ۲۴۵,۱۹۱ دانشجوی کارشناسی ناپیوسته در این دانشگاه‌ها تحصیل می‌کنند

Expend -(هزینه‌ها): میانگین هزینه صرف شده برای هر دانشجو در این نوع دانشگاه‌ها ۱۰,۴۸۶ دلار است که این رقم از میانگین هزینه در دانشگاه‌های غیرخصوصی بیشتر است

Student_percentage -(درصد دانشجویان): حدود ۴۲.۶٪ از کل دانشجویان ثبت‌نام کرده در هر دو نوع دانشگاه در دانشگاه‌های خصوصی ثبت‌نام کرده‌اند

من از تحلیل این داده‌ها نتیجه گرفتم که:

دانشگاه‌های غیرخصوصی در مجموع درخواست‌های بیشتر و ثبت‌نام‌های بیشتری نسبت به دانشگاه‌های خصوصی داشته‌اند.

دانشگاه‌های خصوصی به‌طور میانگین هزینه‌های بیشتری را صرف هر دانشجو می‌کنند.

بیشتر دانشجویان ثبت‌نامی ترجیح داده‌اند که در دانشگاه‌های غیرخصوصی تحصیل کنند تا دانشگاه‌های خصوصی.

۳. **بررسی داده‌های پرت:** شناسایی داده‌های پرت و اطلاعات ناقص و انجام تصمیمات مرتبط با آن‌ها.

۵. **تحلیل مکانی:** اگر داده‌ها مکانی هستند، می‌توانید از نقشه‌ها و تحلیل مکانی برای بررسی الگوها و روابط مکانی استفاده کنید

۶. **تجزیه و تحلیل توزیع‌های احتمالاتی:** این بخش بررسی توزیع داده‌ها و احتمالات مربوط به آنها را شامل می‌شود، مانند توزیع نرمال، توزیع یکنواخت و توزیع دیگر.

۷. **مقایسه داده‌ها:** در این بخش، داده‌ها با یکدیگر مقایسه می‌شوند، از جمله مقایسه توزیع دو یا چند متغیر، مقایسه میانگین یا میانه دو یا چند گروه داده و مقایسه الگوهای مختلف داده‌ها.

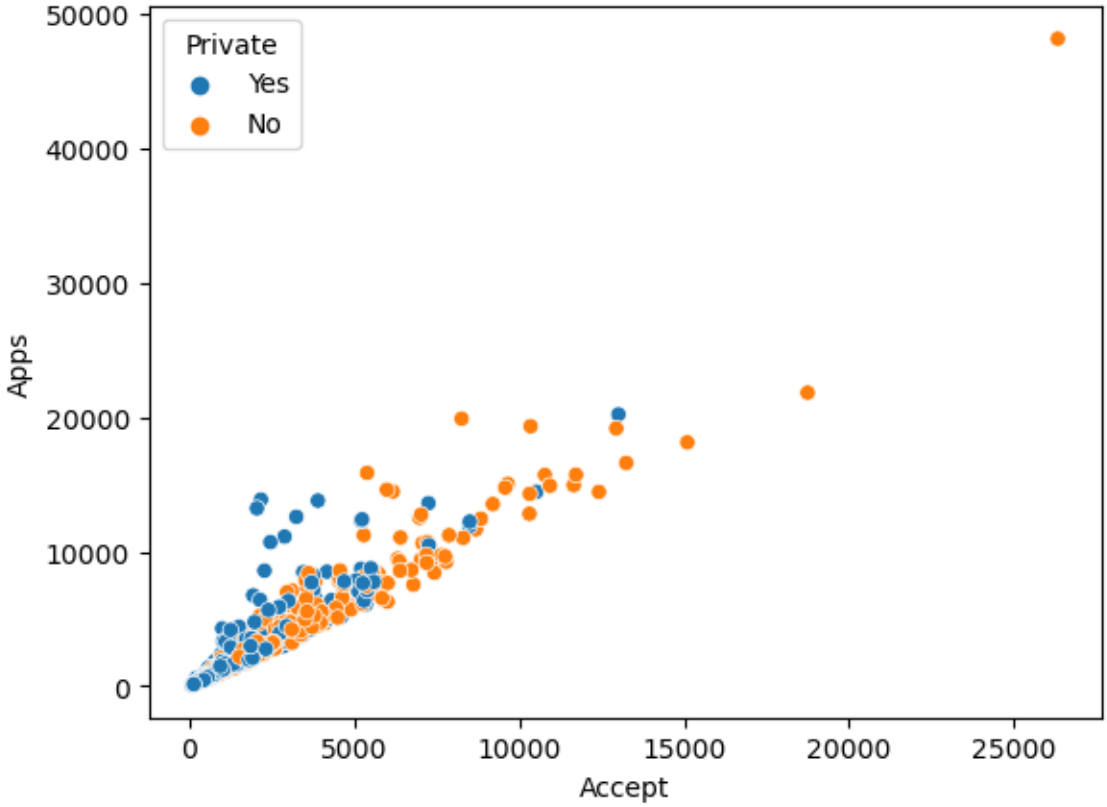
۸.**نمودارها:**(**Visualizaion**): رسم نمودارهای مختلف برای نمایش توزیع و رابطه بین ویژگی‌ها است مثلاً نمودارهای توزیع فراوانی (Histogram) برای نمایش توزیع ویژگی‌ها نمودار پراکندگی (Scatter plot) برای نمایش روابط بین ویژگی‌ها و نمودار جعبه‌ای (Box plot)برای نمایش توزیع ویژگی‌ها بر اساس دسته‌بندی‌های دیگر.

صرفا روی دیتافریم اصلی قبل از پیش پردازش

```
sns.scatterplot(x = 'Accept',y = 'Apps',data = df, hue = 'Private')
```

plt.show()

این کد من برای رسم یک نمودار پراکندگی (scatter plot) است. مقادیر پذیرش (Accept) روی محور افقی (x) و مقادیر درخواست (Apps) روی محور عمودی (y) قرار داده شده‌اند. علاوه بر این، نقاط مختلف بر اساس نوع مؤسسه (خصوصی یا دولتی که در ستون Private مشخص شده است) رنگ‌آمیزی شده‌اند. پس نموداری نمایش داده می‌شود که تفاوت بین دانشگاه‌های خصوصی و دولتی را بر اساس تعداد درخواست‌ها و پذیرش‌ها نشان می‌دهد



این نمودار من یک نمودار پراکندگی است که دو متغیر Accept در محور x و Apps در محور y را نشان می‌دهد هر نقطه نمایانگر یک نقطه داده است که تعداد Accepts متناظر با تعداد Appsاست. نقاط بر اساس اینکه آیا "خصوصی" است یا نه، رنگ‌بندی شده‌اند به طوری که آبی نشان دهنده "بله" (خصوصی بودن) و نارنجی نشان دهنده "خیر" (غیرخصوصی بودن) است.

از توزیع نقاط، به نظر می‌رسد که یک همبستگی مثبت بین Accept و Apps وجود دارد، به این معنی که با افزایش تعدادAccept تعداد Apps نیز به طور معمول افزایش می‌یابد این همبستگی در هر دو گروه خصوصی و غیرخصوصی قابل مشاهده است، زیرا هر دو یک روند نشان می‌دهند که تمرکز نقاط به سمت راست محور x افزایش می‌یابد.

علاوه بر این، نقاط داده‌های خصوصی (بله) و غیرخصوصی (خیر) با یکدیگر تراکم شده‌اند، که نشان می‌دهد اینکه خصوصی بودن یا نبودن، به طور مشخصی داده‌ها را به خوشه‌های متمایز جدا نمی‌کند از نظر Accepts و Apps نقاط پرت نیز وجود دارند، به ویژه تعدادی نقطه برای هر دو دسته که به طور قابل توجهی بالاتر از خوشه‌های اصلی قرار دارند که نشان دهنده مواردی با تعداد Apps بسیار بالا برای تعداد مشخصی Accepts نسبت به روند غیرخصوصی است.

همچنین به نظر می‌رسد که بیشترین تعداد Apps برای تعداد مشخصی Accepts متناظر با یک نقطه داده است که خصوصی نیست (خیر)، همانطور که توسط نقطه نارنجی در بالا و راست نمودار نشان داده شده است.

```
g=sns.FacetGrid(df,hue="Private",palette='coolwarm',height=4,aspect=2)
g = g.map(plt.hist,'Grad.Rate',bins=20,alpha=0.7)
plt.xlabel('Graduation Rate')
plt.ylabel('No of cllg')
plt.legend(['Private','Government'])
plt.show()
```

کدی که نوشتم برای ترسیم هیستوگرام‌هایی است که میزان فارغ‌التحصیلی (Graduation Rate)را بر اساس نوع دانشگاه (خصوصی یا دولتی) نشان می‌دهد.

توضیح کد:

FacetGrid -برای ایجاد یک شبکه از نمودارها استفاده می‌شود که در این‌جا تنها یک شبکه (یک نمودار) با تفکیک Private تولید خواهد شد.

پارامتر palette=coolwarm رنگ‌ها را برای دسته‌بندی‌های مختلف تعریف می‌کند
coolwarmیک تصویر رنگی مرکب است که به صورت گرم و سرد رنگ‌ها را نشان می‌دهد

height=4, aspect=2 -اندازه‌های نمودار را تعریف می‌کند

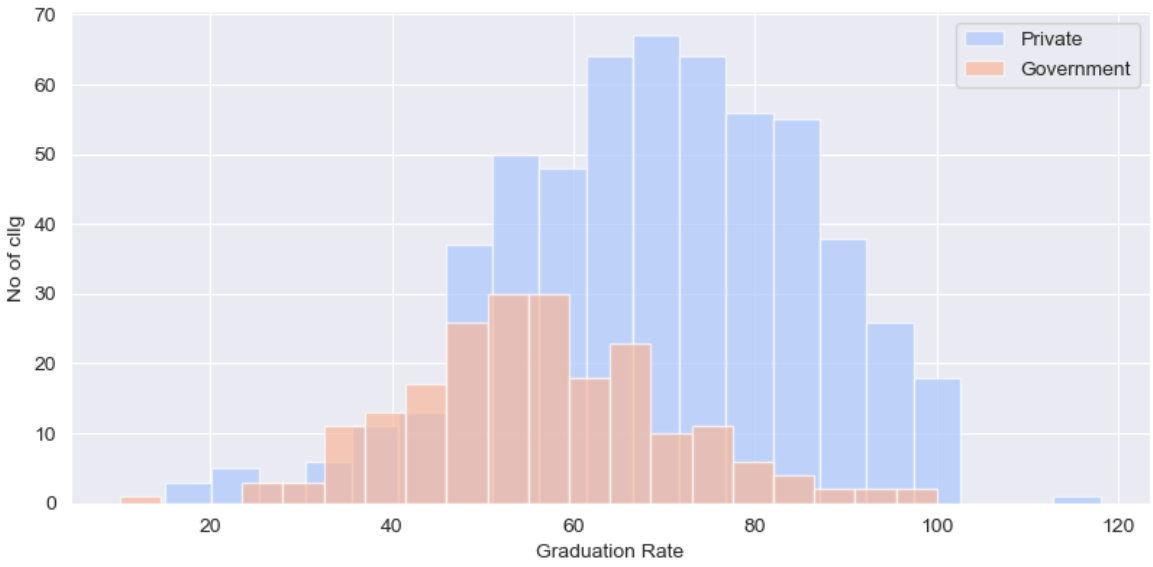
map(plt.hist, Grad.Rate,bins=20,alpha=0.7)-هر بلوک را با استفاده از هیستوگرام و با توجه به ستون Grad.Rate نقشه‌برداری می‌کند bins .تعداد ستون‌های هیستوگرام را تعریف می‌کند و alpha شفافیت بلوک‌ها را تعیین می‌کند.

plt.xlabel -و plt.ylabel برای نام‌گذاری محورهای x و y به کار می‌روند.

plt.legend -افسانه نمودار را می‌سازد و نوع دانشگاه‌ها را نشان می‌دهد.

plt.show() -نمودار را نمایش می‌دهد.

اینجا یک نکته ای است که منطق plt.legend درست است اما Seaborn اتوماتیک متناسب با پارامتر hue در FacetGrid ایجاد می‌کند و نیازی به تعریف دستی نیست اگر بخواهید دستی اضافه کنید باید مطمئن شوید که با داده‌های نمودار هم‌خوانی دارد. در این حالت، اسامی در باید مطابق با داده‌های وجود در ستون Private در df باشد.



این نمودار یک نمودار توزیع داده‌ها به نام هیستوگرام است. این هیستوگرام به توزیع نرخ فارغ التحصیلی برای دو نوع موسسه، یعنی دانشگاه‌های خصوصی و دولتی، مربوط می‌شود.

در اینجا چند مشاهده کلیدی که من از این هیستوگرام کردم:

۱. نرخ فارغ التحصیلی: محور افقی(محور x)نرخ فارغ التحصیلی را نشان می‌دهد که به نظر می‌رسد از حدود ۰ تا ۱۲۰ متغیر است ارزش نرخ فارغ التحصیلی بالاتر از ۱۰۰ غیرمعمول است و ممکن است نشانگر اشتباه در داده یا مقیاس باشد یا ممکن است نشانگر یک روش خاص اندازه‌گیری باشد که می‌تواند بیشتر از ۱۰۰ باشد
۲. تعداد دانشگاه‌ها: محور عمودی(محور y) تعداد دانشگاه‌ها را نشان می‌دهد و با تعداد دانشگاه‌ها نامگذاری شده است.
۳. دانشگاه‌های خصوصی: موسسات خصوصی با نوارهای آبی نمایش داده شده‌اند. توزیع نشان می‌دهد که بیشتر دانشگاه‌های خصوصی نرخ فارغ التحصیلی خود را در بازه ۶۰ تا ۸۰ تجمع داده‌اند و بیشترین فراوانی در این بازه واقع شده است.
۴. دانشگاه‌های دولتی: موسسات دولتی با نوارهای نارنجی نمایش داده شده‌اند. توزیع برای دانشگاه‌های دولتی بیشترین فراوانی خود را در نرخ فارغ التحصیلی کمتر، حدود ۴۰ تا ۶۰،

دارد که نشان می‌دهد تعداد بیشتری از دانشگاه‌های دولتی نرخ فارغ التحصیلی کمتری نسبت به دانشگاه‌های خصوصی دارند

۵. تداخل: همچنین تداخلی بین دو توزیع وجود دارد که نشان می‌دهد برخی از موسسات دولتی نرخ فارغ التحصیلی بالا دارند و برخی از موسسات خصوصی نرخ فارغ التحصیلی پایین‌تری دارند

۶. پراکندگی و انحراف: توزیع دانشگاه‌های خصوصی به نظر می‌رسد پراکندگی بیشتری داشته باشد و کمی به سمت راست انحراف داشته باشد که نشان می‌دهد تعدادی از دانشگاه‌های خصوصی نرخ فارغ التحصیلی بسیار بالایی دارند. توزیع دانشگاه‌های دولتی به نظر می‌رسد چپ‌انحراف‌تر باشد و کمتری از موسسات نرخ فارغ التحصیلی بالایی داشته باشند

این هیستوگرام به طور کلی برای مقایسه عملکرد دانشگاه‌های خصوصی در مقایسه با دانشگاه‌های دولتی از نظر نرخ فارغ التحصیلی استفاده می‌شود.

```
sns.scatterplot(x = 'Grad.Rate', y = 'S.F.Ratio',data= df,hue = 'Private')
plt.xlim(0,100)
```

کد که ارائه دادم یک نقشه پراکندگی می‌سازد که نرخ فارغ‌التحصیلی (معروف به Grad.Rate) را در مقابل نسبت دانشجو به اعضای هیئت علمی (معروف به S.F.Ratio) نشان می‌دهد. داده‌ها بر اساس اینکه آیا دانشگاه خصوصی است یا خیر (ستون Private در داده‌ها) با رنگ‌های مختلفی مشخص شده‌اند با استفاده از (plt.xlim(0,100 ، محدوده محور x نمودار را به مقدار ۰ تا ۱۰۰ تنظیم کرده‌ایم که این کار ما برای نمایش نرخ‌های فارغ‌التحصیلی که معمولاً به صورت درصد ارائه می‌شوند، مناسب است

این نمودار ما پراکندگی دو متغیر را مقایسه می‌کند: نسبت دانشجو به اساتید (S.F.Ratio) در محور y و نرخ فارغ التحصیلی (Grad.Rate) در محور .x نقاط داده‌ها بر اساس یک متغیر دسته‌ای رنگ‌آمیزی شده‌اند که نشان می‌دهد آیا یک مؤسسه خصوصی (بله به رنگ آبی) است یا خصوصی نیست (خیر به رنگ نارنجی).

در ادامه آمدم برخی اطلاعات مهم از نمودار پراکندگی راگفتم:

۱. توزیع نقاط: اکثر نقاط بین مقادیر تقریبی ۱۰ تا ۲۰ برای نسبت دانشجو به اساتید و ۲۰ تا ۸۰ برای نرخ فارغ التحصیلی قرار دارند این نشان می‌دهد که بیشتر مؤسسات موجود در این مجموعه داده نسبت دانشجو به اساتید در این محدوده و به همین ترتیب بین ۲۰٪ و ۸۰٪ از دانشجویان خود را فارغ التحصیل کرده‌اند

۲. مشاهده روند: به نظر نمی‌رسد که یک روند قوی و واضح بین نسبت دانشجو به اساتید و نرخ فارغ التحصیلی وجود داشته باشد. با این حال ممکن است یک افزایش کوچک در نرخ فارغ التحصیلی در صورت کاهش نسبت دانشجو به اساتید (نشان دهنده اندازه کلاس‌های کوچکتر یا کمترین دانشجو برای هر استاد) وجود داشته باشد که این موضوع منطقی به نظر می‌رسد زیرا توجه شخصی بیشتر از استادان می‌تواند بر نرخ فارغ التحصیلی تأثیر مثبتی داشته باشد

۳. مؤسسات خصوصی در مقابل مؤسسات غیرخصوصی: نقاط داده‌های آبی که مؤسسات خصوصی را نشان می‌دهند، بیشتر در سراسر محور نسبت دانشجو به اساتید و نرخ فارغ التحصیلی پخش شده‌اند به عکس، نقاط داده‌های مؤسسات غیرخصوصی (نارنجی) بیشتر خوشه‌ای هستند و به نظر می‌رسد اصلًا بین نسبت دانشجو به اساتید ۱۰ تا ۲۵ و نرخ فارغ التحصیلی زیر ۷۰٪ قرار دارند.

۴. نقاط ناگوار: یک نقطه ناگوار با نرخ فارغ التحصیلی ۱۰۰٪ وجود دارد که غیرمعمول است و ممکن است یک خطا یا نماینده یک مؤسسه بسیار کوچک یا تخصصی باشد. علاوه بر این، چند مؤسسه با نسبت دانشجو به اساتید بسیار بالا، به ویژه یکی از آن‌ها کمی بالاتر از ۴۰ از بقیه داده‌ها متمایز می‌شوند.

۵. مقایسه بین انواع مؤسسات: به طور کلی، مؤسسات خصوصی (نقاط آبی) به نظر می‌رسد دارای محدوده‌ی بالاتری از نرخ فارغ التحصیلی هستند با بیشتر نقاط آبی در نزدیکی انتهای بالای محور نرخ فارغ التحصیلی. پخش نسبت دانشجو به اساتید برای مؤسسات خصوصی نیز به نظر می‌رسد گسترده‌تر باشد، که نشان می‌دهد چگونگی تخصیص منابع اساتید این مؤسسات خصوصی نسبت به دانشجویانشان متنوع‌تر است.

به نظر من مهم است که توجه داشته باشیم که اگرچه این مشاهدات از نمودار پراکندگی قابل استنتاج هستند، اما هیچ علت و معلولیت قابل استنتاج نیست، تنها احتمالات همبستگی ممکن است. برای استنتاجات قطعی‌تر نیاز به تجزیه و تحلیل آماری بیشتری است.

۴)ساخت مدل (Model Building):

Model Building به معنای ساخت مدل است در علوم داده، مدل برای پیش‌بینی و توصیف رفتار داده‌ها به کار می‌رود. برای ساخت یک مدل ابتدا باید داده‌ها را بررسی کرده و تحلیل کرده، سپس به دنبال یافتن الگوهای موجود در داده‌ها هستیم به این منظور، می‌توانیم از الگوریتم‌های یادگیری ماشین استفاده کنیم که با استفاده از داده‌های آموزشی، مدلی را برای پیش‌بینی رفتار داده‌ها ایجاد می‌کنند.

ما از مدل‌های کلاسترینگ در این پروژه استفاده کرده‌ایم

کلاسترینگ (Clustering) یکی از مفاهیم اصلی در یادگیری ماشین بدون نظارت (Unsupervised Learning)است و به فرآیند تقسیم داده‌ها به گروه‌ها یا خوشه‌ها (کلاسترها) بر اساس شباهت‌هایشان اشاره دارد. هدف از کلاسترینگ این است که داده‌هایی که به یک کلاستر تعلق دارند، با یکدیگر شباهت بیشتری داشته باشند تا با داده‌های موجود در کلاسترهای دیگر. به عبارت دیگر، کلاسترینگ برای کشف ساختار طبیعی و گروه‌بندی‌های داخل داده‌ها استفاده می‌شود.

کلاسترینگ کاربردهای مختلفی دارد، از جمله:

تجزیه و تحلیل داده‌های بازاریابی: برای شناسایی گروه‌های مشتری بر اساس علایق و رفتار خرید مشابه.

تجزیه و تحلیل ژنتیکی: گروه‌بندی ژن‌ها یا پروتئین‌ها با عملکردهای مشابه برای مطالعه تکامل و عملکردهای بیولوژیکی.

سازمان‌دهی کامپیوتری: کلاستر کردن اسناد، وب‌سایت‌ها، نقاط داده‌ای سنسورها و غیره بر اساس موضوع یا ویژگی‌های دیگر.

تصویربرداری بخش‌بندی: تقسیم‌بندی تصویر به مناطق مبتنی بر شباهت پیکسل‌ها برای درک بهتر تصویر.

تحلیل شبکه‌های اجتماعی: کلاستر کردن افراد در شبکه‌های اجتماعی بر اساس روابط یا گروه‌های اجتماعی.

چندین روش مختلف برای کلاسترینگ وجود دارد، از جمله:

K-Means Clustering -میانگین‌های مرکزی (Centroids) را محاسبه می‌کند و داده‌ها را بر اساس نزدیک‌ترین میانگین مرکزی کلاستر می‌کند.(من اینو کار کردم)

Hierarchical Clustering -ساختار درختی از داده‌ها را ایجاد می‌کند، که می‌تواند به تقسیم‌بندی خوشه‌ای سطحی یا عمیق منجر شود.

DBSCAN -بر اساس تراکم داده‌ها کار می‌کند و مناطق با تراکم بالای داده را به عنوان یک کلاستر شناسایی می‌کند.

Gaussian Mixture Models -از ترکیبی از توزیع‌های آماری برای مدل‌سازی کلاسترها استفاده می‌کند.

اصولاً، کلاسترینگ برای کشف دانش و الگوهای پنهان در داده‌های بزرگ و پیچیده استفاده می‌شود، که توسط انسان به سختی قابل شناسایی است.

من از k-meanاستفاده کردم

K-Meansیک الگوریتم طبقه‌بندی بدون نظارت (Unsupervised Classification) است که برای کلاستر کردن یا گروه‌بندی داده‌ها به کار می‌رود. این الگوریتم به داده‌ها اجازه می‌دهد تا بر اساس ویژگی‌های مشابهشان، به خودی خود به گروه‌هایی تقسیم شوند. درK-Means ، K نشان دهنده تعداد کلاسترهایی است که می‌خواهیم در داده‌ها بیابیم.

در الگوریتمK-Means ، فرایند به این شکل است که:

۱. تعداد K مرکز (centroid) به‌طور تصادفی انتخاب می‌شود.

۲. هر نقطه داده بر اساس فاصله‌اش به نزدیک‌ترین مرکز اختصاص داده می‌شود، که این کار موجب ساخت K کلاستر می‌شود.

۳. بعد از اختصاص تمام نقاط به کلاسترهای موقت خود، مرکز هر کلاستر دوباره محاسبه و به‌روزرسانی می‌شود.

۴. این مراحل (مرحله ۲ و ۳) تکرار می‌شوند تا زمانی که مراکز دیگر تغییر نکنند یا تغییرات کمتر از یک آستانه مشخص شوند، که در این صورت الگوریتم به پایان می‌رسد.

معیار inertia در K-Means که به صورت مجموع مربعات فاصله‌های هر نقطه از مرکز کلاسترش محاسبه می‌شود به عنوان یک راهنما برای اندازه‌گیری کیفیت کلاسترینگ استفاده می‌شود. انتخاب تعداد مناسب K ضروری است زیرا تعداد کمی از کلاسترها ممکن است به کلاستربندی ناکافی منجر شود و تعداد زیادی کلاستر ممکن است باعث بیش برازش شود.

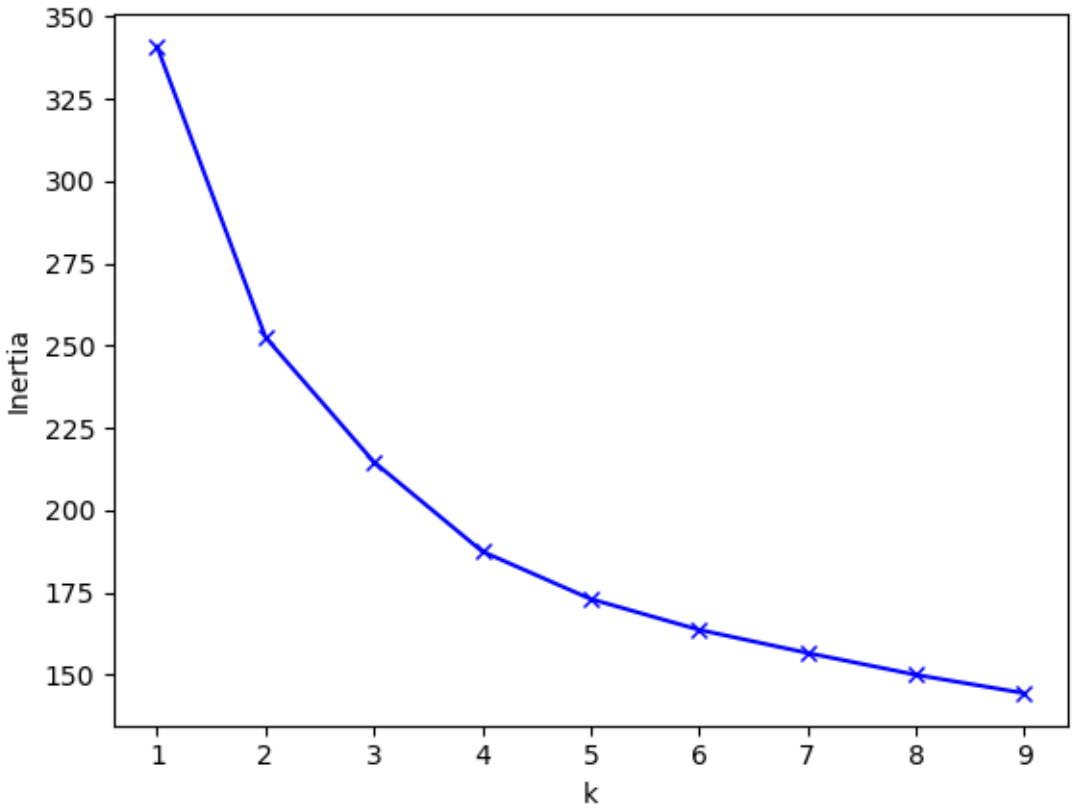
روش آرنج (Elbow Method) معمولاً برای انتخاب تعداد بهینه K به کار می‌رود که در آن یک نمودار از inertia بر اساس مقادیر مختلف K رسم می‌شود و نقطه‌ای که افت درون‌گیری کم می‌شود، به عنوان بهترین ارزش برای K انتخاب می‌شود.

```
from sklearn.cluster import KMeans
inertia = []
K = range(1,10)
for k in K:
 kmeanModel = KMeans(n_clusters=k).fit(features_normal)
 kmeanModel.fit(features_normal)
 inertia.append(kmeanModel.inertia_)
```

کد من فرایند پیاده‌سازی الگوریتم کلاسترینگ K-Means را نشان می‌دهد. این کد برای محاسبه و ذخیره مقادیر inertia (یا درون‌گیری) برای تعداد کلاسترهای مختلف از ۱ تا ۹ است. Inertia شاخصی برای سنجش کیفیت کلاسترینگ است که با مجموع فاصله نقاط تا مرکز کلاستر مربوطه محاسبه می‌شود؛ هرچه این مقدار کمتر باشد، نقاط بیشتر به مراکز کلاسترهای خود نزدیک‌تر هستند و در نتیجه کلاستربندی بهتری داریم این کد ما معمولاً برای یافتن تعداد بهینه کلاسترها (مقدار k مناسب) به کار می‌رود که اغلب با استفاده از روش آرنج (Elbow Method) انجام می‌گیرد که در آن نقطه‌ای که افت inertia کاهش می‌یابد و بعد از آن کاهش بسیار اندکی دارد، به عنوان تعداد بهینه کلاسترها انتخاب می‌شود.

```
plt.plot(K, inertia, 'bx-')
plt.xlabel('k')
plt.ylabel('Inertia')
plt.show()
```

این کد برای رسم نمودار مقادیر درون‌گیری (Inertia) در برابر تعداد مختلف کلاسترهای K استفاده می‌شود. با اجرای این کد، یک نمودار خطی ساخته می‌شود که می‌تواند برای شناسایی بهترین ارزش K با استفاده از روش آرنج مورد استفاده قرار گیرد. نقطه آرنج در نمودار، همان‌جاست که کاهش در درون‌گیری شروع به آهسته شدن می‌کند و معمولاً به عنوان یک انتخاب خوب برای تعداد K در نظر گرفته می‌شود .



این نمودار به نظر من یک نمودار خطی است و ویژگی‌های زیر دارد:

محور x با برچسب k مشخص شده است که به شماره خوشه در زمینه خوشه‌بندی K-means یا الگوریتم خوشه‌بندی دیگری اشاره دارد

محور y با برچسب Inertia مشخص شده است که در زمینه خوشه‌بندی K-means نشان‌دهنده مجموع مربعات داخلی خوشه‌ها است. اینرشیا می‌تواند به عنوان یک معیار برای اندازه‌گیری همبستگی داخلی خوشه‌ها در نظر گرفته شود

نقاط متقاطع آبی در نقاط مختلف روی منحنی قرار دارند که مقادیر اینرشیا در هر k داده شده را نشان می‌دهند

خط این نقاط را به هم متصل می‌کند و روند افزایشی را نشان می‌دهد

این نمودار معمولاً به عنوان نمودار البو شناخته می‌شود و برای تعیین تعداد بهینه خوشه‌ها با یافتن نقطه‌ای که نرخ کاهش به طور قابل توجهی تغییر می‌کند، یعنی البو استفاده می‌شود در نمودار من البو بین k=2 و k=4 قرار دارد که نشان می‌دهد که فراتر از این نقطه، افزایش k به بهبود قابل توجهی در جمعیت خوشه‌ها منجر نمی‌شود و بنابراین k=3 ممکن است یک انتخاب مناسب برای تعداد خوشه‌ها در مجموعه داده‌ای که تحلیل شده است، باشد

این نوع تحلیل معمولاً یک مرحله مهم در تعیین تعداد مناسب خوشه‌ها برای استفاده در الگوریتم‌های تجزیه خوشه‌ها مانند K-means است.

```
kmeans = KMeans(n_clusters=4).fit(features_normal)

labels = pd.DataFrame(kmeans.labels_) #This is where the label output of the KMeans we just ran lives. Make it a dataframe so we can concatenate back to the original data

labeledColleges = pd.concat((features,labels),axis=1)

labeledColleges = labeledColleges.rename({0:'labels'},axis=1)
```

این کد چندین مرحله را برای اجرای الگوریتم K-Means و اتصال برچسب‌های کلاستر به داده‌های اولیه انجام می‌دهد:

۱. kmeans = KMeans(n_clusters=4).fit(features_normal) یک مدل K-Means با ۴ کلاستر ایجاد می‌کند و آن را با استفاده از features_normal (یک مجموعه داده‌های ویژگی که احتمالا نرمال‌سازی شده‌اند) آموزش می‌دهد.
۲. labels = pd.DataFrame(kmeans.labels_) . برچسب‌های تولیدشده توسط K-Means(که هر داده را به یکی از چهار کلاستر اختصاص داده‌اند) را در یک DataFrameجدید قرار می‌دهد تا بتوان آن را به آسانی با داده‌های اصلی ترکیب کرد.

۳. labeledColleges = pd.concat((features,labels),axis=1) این خط کد دو DataFrame یکی که داده‌های اصلی (features) و دیگری که برچسب‌های کلاستر (labels)را در بر دارد، را به هم متصل (concatenate) می‌کند axis=1 نشان‌دهنده اضافه کردن برچسب‌ها بر روی ستون‌ها است (به عبارت دیگر افقی)

۴. labeledColleges = labeledColleges.rename({0:'labels'},axis=1) این خط کد نام ستون حاوی برچسب‌های کلاستر را به labels تغییر می‌دهد. به طور پیش‌فرض پانداس ستون جدید را با عدد ۰ نام‌گذاری می‌کند این خط کد نام ستون را از ۰ به labels تغییر می‌دهد تا قابل فهم و استفاده باشد.

در نهایت، نتیجه اجرای این کد، DataFrame labeledColleges است که شامل داده‌های اصلی به همراه یک ستون جدید به نام labels است که برچسب کلاستر مربوط به هر رکورد (یا دانشگاه در این مثال) را نشان می‌دهد.

tstate	Room.Board	Books	Personal	PhD	Terminal	S.F.Ratio	perc.alumni	Expend	Grad.Rate	Acceptperc	Enrollperc	labels
7440	3300	450	2200	70	78	18.1	12	7041	60	0.742169	0.585227	2
12280	6450	750	1500	29	30	12.2	16	10527	56	0.880146	0.266112	2
11250	3750	400	1165	53	66	12.9	30	8735	54	0.768207	0.306290	2
12960	5450	450	875	92	97	7.7	37	19016	59	0.836930	0.392550	3
7560	4120	800	1500	76	72	11.9	2	10922	15	0.756477	0.376712	2

```
sns.lmplot(x='Top10perc',y='S.F.Ratio',data=labeledColleges,hue='labels',fit_reg=False)
```

این خط کد من برای ایجاد یک نمودار پراکندگی (scatter plot) استفاده می‌کند

توضیح دستور به شرح زیر است :

x=Top10perc -تعیین می‌کند که داده‌های محور افقی (x-axis) از ستون Top10perc در دیتافریم labeledColleges گرفته شود

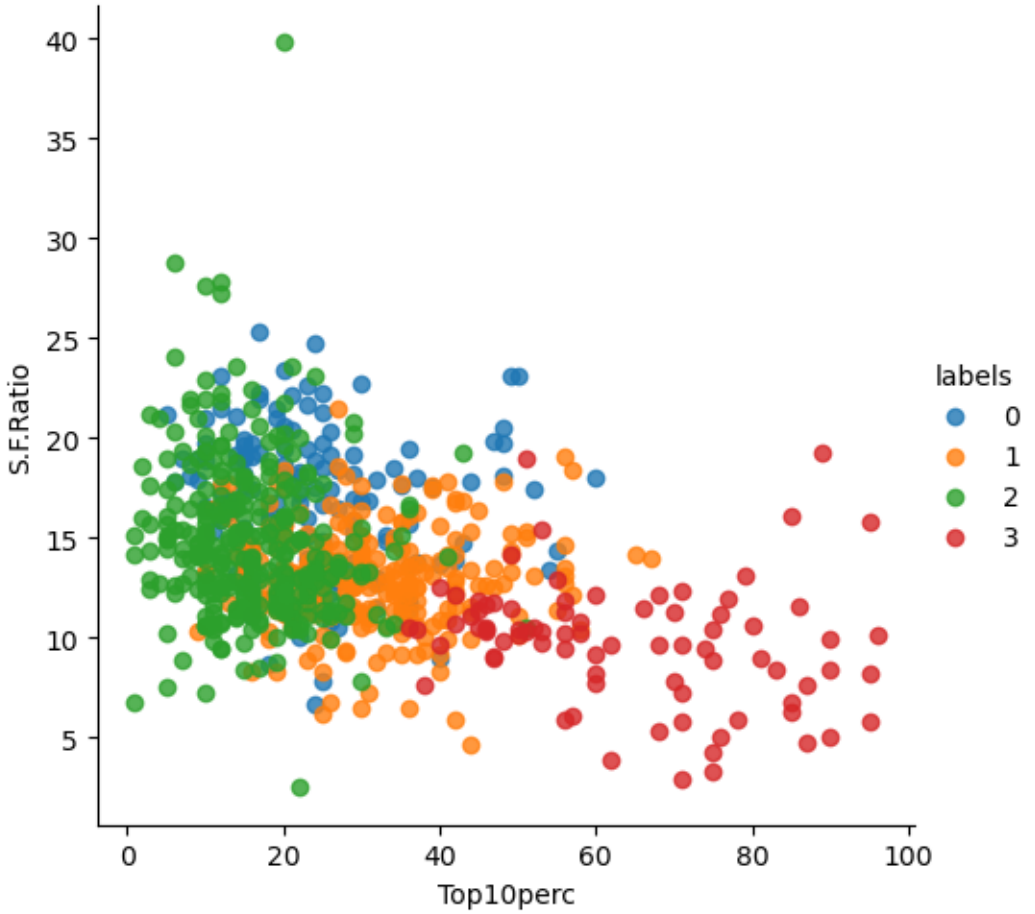
y=S.F.Ratio -تعیین می‌کند که داده‌های محور عمودی (y-axis) از ستون S.F.Ratio در دیتافریم labeledColleges گرفته شود

data=labeledColleges -مشخص می‌کند که داده‌های مورد نیاز برای ساخت نمودار از دیتافریم labeledColleges گرفته شود

hue=labels -این پارامتر رنگ داده‌ها در نمودار پراکندگی را بر اساس برچسب‌های کلاستر تخصیص داده‌شده به هر داده توسط الگوریتم K-Means تعیین می‌کند.

fit_reg=False -این پارامتر به Seaborn می‌گوید که خط رگرسیونی به داده‌ها اضافه نکند.

با اجرای این دستور، یک نمودار پراکندگی ساخته می‌شود که در آن هر نقطه یک دانشکده یا دانشگاه را با مقادیر Top10perc در محور x و S.F.Ratio در محور y مشخص می‌کند. نقاط با استفاده از متغیر labels که برچسب‌های کلاستر را نشان می‌دهد بر اساس چهار کلاستر مختلف رنگ‌بندی خواهند شد



این نمودار یک نمودار پراکندگی است که شامل یک سری نقاط است که بر روی یک سیستم مختصات رسم شده‌اند هر نقطه نشان‌دهنده یک مشاهده داده با دو متغیر است. محور x با عنوان Top10perc است و از ۰ تا ۱۰۰ مقدار گرفته است محور y با عنوان S.F.Ratio است و از تقریباً ۰ تا ۴۵ مقدار گرفته است نقاط روی نمودار پراکندگی به طور مختلف رنگ‌آمیزی شده‌اند تا دسته‌ها یا گروه‌های مختلف را نشان دهند همانطور که در یک علامت برچسب‌ها با چهار دسته‌بندی شماره‌گذاری شده از ۰ تا ۳ مشخص شده است هر رنگ یکی از برچسب‌ها را نشان می‌دهد: قرمز برای ۰، سبز برای ۱، نارنجی برای ۲ و آبی برای ۳

پراکندگی نقاط نشان‌دهنده وجود یک سطحی از همبستگی بین متغیرهای Top10perc و S.F.Ratio است با یک روند کلی که هرچه Top10perc افزایش می‌یابد S.F.Ratio دارای یک روند کمی به سمت بالا است به نظر می‌رسد که یک تراکم نسبتاً چگال وجود دارد که

Top10perc در محدوده میانی است و S.F.Ratio در محدوده پایین تا میانی است این خوشه‌ها به نظر می‌رسد تا حدی توسط رنگ از هم جدا شده‌اند که نشان می‌دهد هر گروه ممکن است ویژگی‌های متمایز در داخل مجموعه داده را نشان دهد

```
sns.pairplot(labeledColleges,hue='labels')
```

در کتابخانه Seaborn پایتون تابع pairplot برای رسم نمودارهای پراکندگی چندتایی بین همه جفت متغیرهای عددی در یک DataFrame و همچنین توزیع تک متغیره (univariate distributions)برای هر متغیر به تنهایی استفاده می‌شود

وقتی پارامتر hue مشخص شود نمودارهای پراکندگی و هیستوگرام‌ها بر اساس دسته‌بندی مشخص‌شده توسط ویژگی مشخص‌شده در hue رنگ‌بندی می‌شوند در این مورد، labelsکه از الگوریتم K-Means به دست آمده و برچسب هر نمونه را به عنوان یکی از کلاسترها نشان می‌دهد، به عنوان پارامتر hue استفاده می‌شود

بنابراین، اجرای sns.pairplot(labeledColleges, hue='labels') یک شبکه نموداری ایجاد می‌کند که جفت‌های هر دو ویژگی عددی موجود در دیتافریم labeledColleges را نشان می‌دهد. نقاط در نمودارهای پراکندگی بر اساس برچسب کلاسترشان رنگ‌بندی می‌شوند و هیستوگرام‌های روی قطر اصلی توزیع هر ویژگی را به تفکیک کلاستر نشان می‌دهند

این نوع نمودار برای اکتشاف ارتباطات بالقوه و الگوهای موجود در داده‌ها با توجه به تفاوت‌های بین کلاسترها مفید است

K-meanدیگر با تارگت متفاوت

```
x = df.drop(['Unnamed: 0','Private','Personal'],axis = 1)
kmean = KMeans(n_clusters= 2)
```

```
kmean.fit(x)
```

دستوراتی که گفتم مراحل اولیه برای اجرای الگوریتم خوشه‌بندی K-Means روی یک مجموعه داده را نشان می‌دهد.

در اینجا هر خط چه کاری انجام می‌دهد:

۱. x = df.drop(['Unnamed: 0','Private','Personal'], axis = 1) این دستور از df سه ستون Unnamed: 0', Private, و Personal را حذف می‌کند این ستون‌ها ممکن است برای مدل خوشه‌بندی مفید نباشند یا داده‌هایی باشند که به صورت ناخواسته وجود دارند.

۲. kmean = KMeans(n_clusters= 2) با این دستور یک نمونه از کلاس KMeansاز کتابخانه scikit-learn ایجاد می‌شود، با پارامتر n_clusters= 2 که تعیین می‌کند الگوریتم باید داده‌ها را به دو خوشه تقسیم کند

۳. kmean.fit(x) این خط کد الگوریتم K-Means را روی داده‌های x هدایت می‌کند یعنی الگوریتم سعی می‌کند ساختار دیتا را تجزیه و تحلیل کرده و داده‌ها را به دو گروه تقسیم کند، بر اساس ویژگی‌هایی که در دیتا وجود دارند

به این طریق ما می‌توانید پتانسیل پنهان در داده‌های خود را کشف کنید، مانند گروه‌بندی دانشگاه‌ها بر اساس ویژگی‌های مختلف. پس از اجرا شما خواهید توانست برچسب‌ها یا نتایج الگوریتم را در متغیر kmean.labels ببینید که برچسب هر رکورد را بر اساس خوشه‌ای که به آن تعلق دارد نشان می‌دهد.

```
identified_cluster = kmean.fit_predict(x)
df['cluster'] = identified_cluster
```

در این دو خط کد من یک مدل خوشه‌بندی K-Means بر روی مجموعه داده‌ها اجرا و نتایج آن به مجموعه داده اصلی اضافه کرد

توضیحات کد :

۱. identified_cluster = kmean.fit_predict(x) این خط الگوریتم K-Means را بر روی دیتافریم x اجرا کرده و خروجی را به صورت یک آرایه ذخیره می‌کند که حاوی برچسب خوشه‌ای می‌باشد که هر نمونه (ردیف) به آن تعلق دارد تابع fit_predict هم اجرای مدل و هم برگرداندن برچسب‌های خوشه را ترکیب کرده است بنابراین بعد از این دستور، هر داده در x به یکی از دو خوشه (چون `n_clusters=2` بود) اختصاص داده شده است

۲. df['cluster'] = identified_clusterدر این خط کد من برچسب‌های خوشه تشخیص داده شده به مجموعه داده اصلی df اضافه کردم یک ستون جدید به نام clusterایجاد شده و هر ردیف در df برچسب خوشه‌ای که به آن اختصاص داده شده را دریافت می‌کند

به این ترتیب شما می‌توانید ببینید که هر نمونه در مجموعه داده اصلی، به کدام یک از دو خوشه تعیین شده توسط الگوریتم K-Means تعلق دارد این اطلاعات را می‌توان برای تحلیل‌های بیشتر استفاده کرد مانند مقایسه ویژگی‌های مختلف دیتا بین خوشه‌ها

```
from sklearn import metrics

results = metrics.confusion_matrix(y_true =
df['Private'].map({'Yes':0,'No':1}),y_pred = df['cluster'])

cm_display = metrics.ConfusionMatrixDisplay(confusion_matrix =
results,display_labels=['Private','Government'])

cm_display.plot()

plt.show()
```

این بخش از کد من یک ماتریس درهم‌ریختگی (confusion matrix) ایجاد می‌کند که یکی از روش‌های ارزیابی عملکرد مدل‌های طبقه‌بندی است عملکرد الگوریتم K-Means که پیشتر اجرا شده است را بر اساس یک معیار دسته‌بندی واقعی که در این مورد برچسب‌های Private است ارزیابی می‌کند

توضیحات کد:

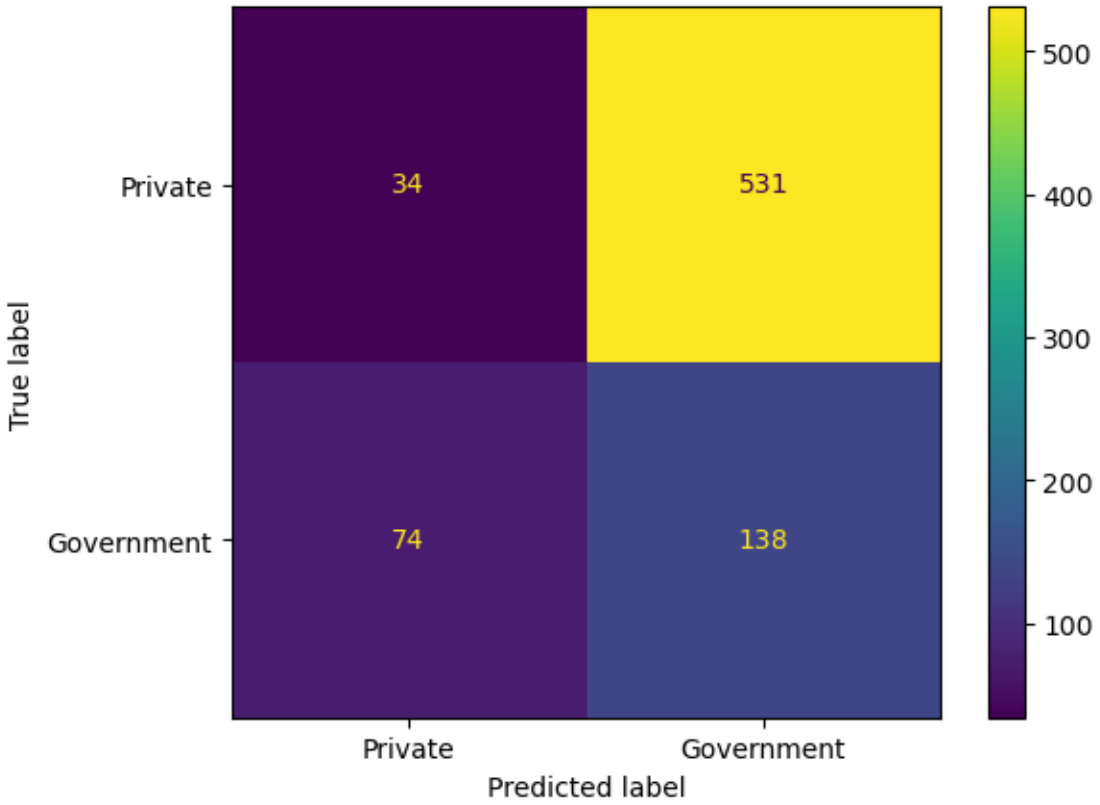
۱. results = metrics.confusion_matrix(...) اینجا یک ماتریس درهم‌ریختگی با استفاده از پیش‌بینی‌های حاصل از مدل K-Means (df['cluster']) و مقادیر واقعی یا مورد انتظار (df['Private'].map({'Yes':0,'No':1})) محاسبه می‌شود مقادیر Yes و Noدر ستون Private به ۰ و ۱ نگاشت داده شده‌اند تا با برچسب‌های عددی خروجی الگوریتم K-Means مطابقت داشته باشند

۲. cm_display = metrics.ConfusionMatrixDisplay(...) یک شیء نمایش ماتریس درهم‌ریختگی با استفاده از ماتریس درهم‌ریختگی حاصل (results) و برچسب‌هایی که برای دو کلاس (Private, Government) استفاده می‌شوند ایجاد می‌کند

۳. cm_display.plot() این دستور متد plot() را روی شیء نمایش ماتریس درهم‌ریختگی فراخوانی می‌کند تا نمودار ماتریس درهم‌ریختگی را ترسیم کند

۴. plt.show() این خط کد استفاده می‌شود تا نمودار ماتریس درهم‌ریختگی را نمایش دهد

نهایتاً، این بخش از کد نتایج خوشه‌بندی را به صورت بصری ارائه می‌دهد و به شما امکان می‌دهد بررسی کنید که چگونه الگوریتم خوشه‌بندی نسبت به داده‌های برچسب‌دار واقعی عملکرد داشته است این امر به ویژه مفید است وقتی برچسب‌ها برای دو نوع دانشگاه خصوصی و دولتی وجود دارد و مایل هستید ببینید نتایج خوشه‌بندی چگونه با این دسته‌بندی مطابقت دارد.



این یک ماتریس ابهام است که به طور معمول در یادگیری ماشین برای ارزیابی عملکرد مدل‌های طبقه‌بندی استفاده می‌شود.

بر اساس اطلاعات ارائه شده من این تحلیل را میکنم:

۱. محور افقی نمایانگر برچسب‌های پیش‌بینی شده توسط مدل است، با دو دسته خصوصی و دولتی
۲. محور عمودی نمایانگر برچسب‌های واقعی است، با همان دو دسته خصوصی و دولتی
۳. ماتریس ابهام از چهار سلول تشکیل شده است هر کدام نمایانگر تعداد پیش‌بینی‌ها در مقایسه با طبقه‌بندی‌های واقعی هستند:

- سلول بالا و سمت چپ: عدد ۳۴ نمایانگر تعداد نمونه‌هایی است که در واقع خصوصی بودند اما به اشتباه به عنوان خصوصی توسط مدل پیش‌بینی شدند.
- سلول بالا و سمت راست: عدد ۵۳۱ نمایانگر تعداد نمونه‌هایی است که به درستی به عنوان "خصوصی" پیش‌بینی شدند (مثبت‌های واقعی برای کلاس خصوصی).

- سلول پایین و سمت چپ: عدد ۷۴ نمایانگر تعداد نمونه‌هایی است که در واقع دولتی بودند اما به اشتباه به عنوان خصوصی توسط مدل پیش‌بینی شدند (مثبت‌های اشتباه برای کلاس خصوصی و منفی‌های اشتباه برای کلاس دولتی)
- سلول پایین و سمت راست: عدد ۱۳۸ نمایانگر تعداد نمونه‌هایی است که به درستی به عنوان دولتی پیش‌بینی شدند (مثبت‌های واقعی برای کلاس دولتی)

شدت رنگ و نوار رنگی در سمت راست نمایانگر تعداد نمونه‌ها برای هر سلول است، به طوری که رنگ‌های تیره نمایانگر تعداد بیشتری از نمونه‌ها هستند

از این ماتریس ابهام می‌توانیم نتیجه بگریم که مدل بیشتر نمونه‌ها را به درستی به عنوان خصوصی پیش‌بینی کرده است تا دولتی با این حال، مواردی هم وجود دارد که هر دو برچسب خصوصی و دولتی را اشتباه طبقه‌بندی کرده است. این اعداد می‌توانند برای محاسبه معیارهای عملکرد مختلف مانند دقت، دقت پیش‌بینی، بازخوانی و امتیاز F1 استفاده شوند.

بر اساس ماتریس ابهام که ارائه شد می‌توانیم برخی از معیارهای ارزیابی عملکرد مدل را محاسبه کنیم. اما بحث بر اساس این اطلاعات به میزان کافی نیست و نیاز به بررسی عمیق‌تری از عملکرد مدل و اطلاعات بیشتری از داده‌ها و شرایط آزمایش داریم به عنوان مثال، ممکن است مدل در یکی از دسته‌ها به طور قابل توجهی بهتر عمل کرده باشد و در دیگر دسته نتایج ضعیف‌تری داشته باشد. همچنین، نیاز به مقایسه عملکرد مدل با مدل‌های دیگر و ارزیابی تاثیرپذیری آن از تغییرات داده‌ها و پارامترهای مدل وجود دارد. بنابراین، برای ارزیابی کامل عملکرد مدل، نیاز به یک بررسی جامع‌تر و دقیق‌تر داریم.

متشکرکه تا اینجا من را همرایم کردید 🙏

پایان