به نام خداوند جان و خرد

دانشگاه تهران

دانشکدگان فنی

دانشکده مهندسی برق و کامپیوتر

یادگیری ماشین

تمرین شماره ۵

نام و نام خانوادگی: علی خرم فر

شماره دانشجویی: ۸۱۰۱۰۲۱۲۹

تیرماه ۱۴۰۳

فهرست مطالب

۱_پاسخ سوال ۱

پاسخ اسکن‌شده پیوست شد.

۲_ پاسخ سوال ۲

۲-۱_ Model Selection و Model Assessment

در یادگیری ماشین، انتخاب مدل (Model Selection) و ارزیابی مدل (Model Assessment) دو مرحله بسیار مهم هستند که هر یک هدف و روش‌های خاص خود را دارند. در ادامه هردوی آن‌ها را بررسی خواهیم کرد.

انتخاب مدل (Model Selection)

انتخاب مدل فرآیندی است که در آن از میان چندین مدل مختلف، بهترین مدل بر اساس عملکرد آن‌ها بر روی داده‌های آموزشی و اعتبارسنجی انتخاب می‌شود. هدف اصلی انتخاب مدل این است که مدلی را بیابیم که بهترین عملکرد را بر روی داده‌های جدید و دیده‌نشده داشته باشد. در این فرآیند، از روش‌هایی مانند Cross-Validation استفاده می‌شود. در Cross-Validation ، داده‌ها به k بخش تقسیم می‌شوند و مدل روی k-1 بخش آموزش داده می‌شود و بر روی بخش باقی‌مانده ارزیابی می‌شود. این فرآیند k بار تکرار می‌شود و میانگین نتایج به عنوان عملکرد نهایی مدل در نظر گرفته می‌شود. علاوه بر این، معیارهای اطلاعاتی مانند AIC و BIC نیز برای مقایسه مدل‌ها بر اساس پیچیدگی و احتمال آن‌ها به کار می‌روند. همچنین، استفاده از مجموعه داده‌های اعتبارسنجی به تنظیم بهینه پارامترهای مدل کمک می‌کند.

پس از آماده‌سازی داده‌ها، مجموعه‌ای از مدل‌های یادگیری ماشین که احتمال می‌رود بتوانند مسأله را به خوبی حل کنند، انتخاب می‌شوند. این مدل‌ها ممکن است شامل رگرسیون خطی، درخت تصمیم‌گیری، جنگل تصادفی و شبکه‌های عصبی باشند. داده‌ها به مجموعه‌های آموزشی و اعتبارسنجی تقسیم می‌شوند، به طوری که مدل‌ها با استفاده از داده‌های آموزشی آموزش داده شده و با داده‌های اعتبارسنجی ارزیابی اولیه می‌شوند. معیارهای مختلفی مانند دقت، نرخ خطا و معیارهای دیگر برای ارزیابی عملکرد مدل‌ها استفاده می‌شود. سپس هایپرپارامترهای مدل‌ها با استفاده از روش‌هایی مانند جستجوی شبکه‌ای (Grid Search) یا جستجوی تصادفی (Random Search) تنظیم می‌شوند تا بهترین تنظیمات ممکن برای هر مدل پیدا شود.

در نهایت، مدلی که بهترین عملکرد را روی داده‌های اعتبارسنجی دارد، به عنوان مدل نهایی انتخاب می‌شود. این مدل برای آموزش نهایی آماده می‌شود و تمامی داده‌های آموزشی (ترکیبی از داده‌های آموزشی و اعتبارسنجی قبلی) برای آموزش مجدد آن استفاده می‌شوند تا از حداکثر اطلاعات موجود بهره‌برداری شود. این فرآیند انتخاب مدل به ما اطمینان می‌دهد که بهترین مدل ممکن برای داده‌های ما انتخاب شده است و آماده برای ارزیابی نهایی است.

ارزیابی مدل (Model Assessment)

ارزیابی مدل فرآیندی است که در آن عملکرد مدل نهایی انتخاب شده بر روی داده‌های تست مستقل بررسی می‌شود. هدف این مرحله، تخمین خطای پیش‌بینی مدل و ارزیابی توانایی آن در تعمیم به داده‌های جدید است. برای انجام این ارزیابی، مدل بر روی مجموعه داده‌های تست که در مراحل قبلی استفاده نشده‌اند، آزموده می‌شود. معیارهای عملکرد مختلفی مانند خطای میانگین مربعات (MSE)، دقت (Accuracy)، فراخوانی (Recall) و F1 Score برای ارزیابی مدل به کار می‌روند. این فرآیند تضمین می‌کند که مدل نهایی بر روی داده‌های جدید و دیده‌نشده عملکرد مناسبی خواهد داشت.

ابتدا مدل نهایی انتخاب شده با استفاده از تمامی داده‌های آموزشی آموزش داده می‌شود تا از تمامی اطلاعات موجود بهره‌برداری شود و مدل بتواند بهترین عملکرد خود را ارائه دهد. سپس داده‌های آزمون نیز مانند داده‌های آموزشی پیش‌پردازش می‌شوند تا آماده برای ارزیابی باشند. مدل نهایی آموزش دیده برای پیش‌بینی نتایج بر روی داده‌های آزمون استفاده می‌شود. این مرحله شامل استفاده از مدل برای انجام پیش‌بینی‌هایی بر اساس ورودی‌های داده‌های آزمون است.

عملکرد مدل با استفاده از پیش‌بینی‌های انجام شده و نتایج واقعی داده‌های آزمون ارزیابی می‌شود. معیارهای مختلفی مانند دقت، نرخ خطا، دقت و بازخوانی (Precision and Recall)، ماتریس درهم‌ریختگی (Confusion Matrix) و خطای مطلق میانگین (Mean Absolute Error) برای ارزیابی عملکرد مدل استفاده می‌شوند.

تحلیل نتایج ارزیابی به ما کمک می‌کند تا نقاط قوت و ضعف مدل را بشناسیم. این تحلیل‌ها می‌توانند به بهبودهای آتی مدل کمک کنند و نشان دهند که آیا مدل نیاز به تنظیمات بیشتر یا اصلاح دارد. هدف نهایی ارزیابی مدل این است که اطمینان حاصل کنیم مدل انتخاب شده توانایی عملکرد مطلوب در مواجهه با داده‌های جدید و دیده نشده را دارد و می‌تواند به طور مؤثر در دنیای واقعی مورد استفاده قرار گیرد. این فرآیند ارزیابی به ما اجازه می‌دهد تا عملکرد مدل را در شرایط واقعی تست کنیم و از کیفیت و قابلیت اعتماد مدل اطمینان حاصل کنیم.

تفاوت Model Selection و Model Assessment

تفاوت‌های بین انتخاب مدل (Model Selection) و ارزیابی مدل (Model Assessment) در چندین جنبه مختلف، از جمله داده‌های استفاده‌شده، هدف فرآیند، روش‌ها و خروجی‌ها، قابل بررسی است:

داده‌های استفاده‌شده:

در فرآیند انتخاب مدل، از داده‌های آموزشی و اعتبارسنجی استفاده می‌شود. داده‌های آموزشی برای آموزش مدل‌های مختلف و داده‌های اعتبارسنجی برای ارزیابی و انتخاب بهترین مدل استفاده می‌شوند. این داده‌ها معمولاً از مجموعه داده‌های اصلی استخراج می‌شوند و به‌صورت تصادفی به دو بخش تقسیم می‌شوند تا مدل‌ها بتوانند با داده‌های آموزشی یاد بگیرند و با داده‌های اعتبارسنجی مورد ارزیابی قرار گیرند. در مقابل، در فرآیند ارزیابی مدل، داده‌های آزمون که قبلاً در فرآیند انتخاب مدل استفاده نشده‌اند، برای ارزیابی نهایی عملکرد مدل استفاده می‌شوند. این داده‌های آزمون به‌عنوان داده‌های دیده‌نشده عمل می‌کنند و نمایانگر توانایی مدل در تعمیم به داده‌های جدید هستند.

هدف فرآیند:

هدف انتخاب مدل، یافتن بهترین مدل یا تنظیمات مدل است که بهترین عملکرد را در داده‌های آموزشی و اعتبارسنجی داشته باشد. این شامل انتخاب نوع مدل، تنظیم هایپرپارامترها و حتی انتخاب ویژگی‌ها است. به عبارت دیگر، انتخاب مدل فرآیندی است که به کمک آن مدل بهینه‌ای برای داده‌های خاص پیدا می‌شود. در مقابل، هدف ارزیابی مدل تخمین عملکرد واقعی مدل انتخاب شده بر روی داده‌های دیده نشده است. این ارزیابی نشان می‌دهد که مدل در مواجهه با داده‌های جدید چقدر دقیق و قابل اعتماد است و به ما اطمینان می‌دهد که مدل می‌تواند به‌طور مؤثر در دنیای واقعی استفاده شود.

روش‌ها

روش‌های مورد استفاده در انتخاب مدل شامل تقسیم داده‌ها به مجموعه‌های آموزشی و اعتبارسنجی، آموزش مدل‌های مختلف، ارزیابی عملکرد آن‌ها با استفاده از داده‌های اعتبارسنجی، و تنظیم هایپرپارامترها می‌باشد. این روش‌ها به مدل‌ها اجازه می‌دهند تا با داده‌های آموزشی یاد بگیرند و سپس با داده‌های اعتبارسنجی ارزیابی شوند تا بهترین مدل انتخاب شود. در مقابل، روش‌های ارزیابی مدل شامل استفاده از مدل نهایی آموزش دیده برای پیش‌بینی نتایج بر روی داده‌های آزمون و ارزیابی عملکرد آن با استفاده از معیارهای مختلفی مانند دقت، نرخ خطا، دقت و فراخوانی (Precision and Recall)، ماتریس درهم‌ریختگی (Confusion Matrix) و خطای مطلق میانگین (Mean Absolute Error) است. این ارزیابی‌ها نشان‌دهنده توانایی مدل در پیش‌بینی دقیق و عملکرد قابل اعتماد در مواجهه با داده‌های جدید هستند.

خروجی‌ها

خروجی انتخاب مدل یک مدل نهایی انتخاب شده است که بهترین عملکرد را در داده‌های اعتبارسنجی داشته است. این مدل به‌عنوان مدل بهینه‌ای برای مسئله مورد نظر در نظر گرفته می‌شود و آماده برای آموزش نهایی با استفاده از تمامی داده‌های آموزشی است. در مقابل، خروجی ارزیابی مدل مجموعه‌ای از معیارهای عملکرد است که نشان‌دهنده کارایی مدل در داده‌های جدید و دیده نشده می‌باشد. این معیارها به ما کمک می‌کنند تا بفهمیم مدل چقدر دقیق و قابل اعتماد است و آیا نیاز به بهبودها یا تنظیمات بیشتر دارد یا خیر.

به‌طور کلی، انتخاب مدل فرآیندی است که به کمک آن بهترین مدل از میان چندین مدل کاندید انتخاب می‌شود و شامل استفاده از داده‌های آموزشی و اعتبارسنجی است. هدف اصلی آن یافتن مدل بهینه برای داده‌های خاص است. در مقابل، ارزیابی مدل فرآیندی است که به کمک آن عملکرد واقعی مدل انتخاب شده با استفاده از داده‌های دیده نشده ارزیابی می‌شود. هدف اصلی آن تخمین توانایی مدل در مواجهه با داده‌های جدید و اطمینان از قابلیت اعتماد و دقت آن در دنیای واقعی است. این دو فرآیند با اینکه مرتبط و مکمل یکدیگر هستند، اما نقش‌ها و اهداف متفاوتی در توسعه مدل‌های یادگیری ماشین دارند.

۲-۲_ الگوریتم‌های Probabilistic و Resampling در Model Selection

الگوریتم‌های Probabilistic

الگوریتم‌های Probabilistic در انتخاب مدل از اصول و تئوری‌های احتمالاتی برای انتخاب بهترین مدل از میان مدل‌های مختلف استفاده می‌کنند. این روش‌ها بر اساس احتمال وقوع مدل‌های مختلف و معیارهای اطلاعاتی که پیچیدگی و دقت مدل را در نظر می‌گیرند، تصمیم‌گیری می‌کنند. یکی از شناخته‌شده‌ترین این معیارها، Akaike Information Criterion (AIC) است که به دنبال مدلی می‌گردد که تعادل مناسبی بین پیچیدگی و دقت داشته باشد. AIC با استفاده از تعداد پارامترهای مدل و حداکثر مقدار احتمال به دست آمده از مدل، مدلی را انتخاب می‌کند که کمترین مقدار AIC را دارد. این به این معناست که مدلی که توانسته بهترین توازن بین تطابق با داده‌ها و ساده بودن را حفظ کند، انتخاب می‌شود.

دیگر معیار مهم در این دسته Bayesian Information Criterion (BIC) است که مشابه AIC عمل می‌کند اما جریمه بیشتری برای پیچیدگی مدل قائل می‌شود. BIC براساس تئوری اطلاعات بیزین کار می‌کند و مدلی را انتخاب می‌کند که بیشترین احتمال را با توجه به داده‌های مشاهده شده داشته باشد. این معیار به‌ویژه در مواردی که می‌خواهیم مدلی با پیچیدگی کمتر و تعمیم‌پذیری بهتر داشته باشیم، مفید است.

روش دیگری که در این دسته قرار می‌گیرد، Bayesian Model Averaging (BMA) است. به جای انتخاب یک مدل واحد، BMA از چندین مدل استفاده می‌کند و پیش‌بینی‌ها را براساس احتمال وقوع هر مدل ترکیب می‌کند. این روش با استفاده از توزیع‌های احتمالی، میانگین‌گیری از نتایج مدل‌های مختلف را انجام می‌دهد و بنابراین به کاهش عدم قطعیت و افزایش دقت پیش‌بینی کمک می‌کند.

یکی از مزایای اصلی الگوریتم‌های Probabilistic این است که به طور مستقیم به پیچیدگی مدل توجه می‌کنند و سعی می‌کنند مدلی را انتخاب کنند که نه تنها بهترین تطابق را با داده‌ها دارد، بلکه ساده‌ترین مدل ممکن نیز باشد. با این حال، این روش‌ها به دلیل نیاز به محاسبات پیچیده و زمان‌بر ممکن است برای مجموعه داده‌های بسیار بزرگ یا مدل‌های بسیار پیچیده کارایی نداشته باشند. علاوه بر این، این الگوریتم‌ها نیاز به فرضیات قوی درباره توزیع داده‌ها و مدل‌ها دارند که ممکن است همیشه صحیح نباشند.

الگوریتم‌های Resampling

گوریتم‌های Resampling از تکنیک‌های بازنمونه‌گیری داده‌ها برای ارزیابی و انتخاب بهترین مدل استفاده می‌کنند. این روش‌ها با تقسیم داده‌ها به مجموعه‌های مختلف و انجام مکرر آموزش و ارزیابی مدل‌ها بر روی این مجموعه‌ها، عملکرد مدل‌ها را بررسی می‌کنند. یکی از پرکاربردترین روش‌های Resampling، Cross-Validation (اعتبارسنجی متقابل) است. در این روش، داده‌ها به چندین بخش (معمولاً k بخش) تقسیم می‌شوند و مدل به صورت متوالی بر روی ترکیبات مختلف این بخش‌ها آموزش و ارزیابی می‌شود. به طور متداول، در k-fold Cross-Validation، داده‌ها به k بخش مساوی تقسیم می‌شوند، و هر بار یکی از این بخش‌ها به عنوان مجموعه آزمون و بقیه به عنوان مجموعه آموزشی استفاده می‌شوند. این فرآیند k بار تکرار می‌شود و در نهایت میانگین نتایج به‌دست آمده به عنوان عملکرد نهایی مدل در نظر گرفته می‌شود.

روش دیگر در دسته Resampling، Bootstrap است. در این روش، از داده‌های موجود به‌طور مکرر نمونه‌گیری با بازگشت انجام می‌شود و مدل‌ها بر روی این نمونه‌ها آموزش و ارزیابی می‌شوند. Bootstrap به ما اجازه می‌دهد تا توزیع عملکرد مدل‌ها را بر اساس نمونه‌های مختلف داده‌ها بررسی کنیم و تخمین‌های پایدارتری از عملکرد مدل به‌دست آوریم.

یکی از مزایای اصلی روش‌های Resampling این است که نیاز به فرضیات کمتری درباره توزیع داده‌ها و مدل‌ها دارند و می‌توانند در مواردی که داده‌ها توزیع غیرمعمول یا پیچیده دارند، کارایی بالایی داشته باشند. این روش‌ها به دلیل بازنمونه‌گیری مکرر، تخمین‌های پایدارتری از عملکرد مدل ارائه می‌دهند. با این حال، این روش‌ها نیز زمان‌بر هستند و ممکن است برای مجموعه داده‌های بسیار بزرگ، نیاز به منابع محاسباتی زیادی داشته باشند.

در مجموع، الگوریتم‌های Resampling به دلیل سادگی و انعطاف‌پذیری بیشتر معمولاً در بسیاری از کاربردهای عملی ترجیح داده می‌شوند، در حالی که الگوریتم‌های Probabilistic ممکن است در شرایطی که دقت بالا و توجه به پیچیدگی مدل اهمیت دارد، مورد استفاده قرار گیرند. انتخاب بین این دو دسته بستگی به نیازهای خاص مسئله، محدودیت‌های زمانی و محاسباتی، و میزان دانش درباره توزیع داده‌ها دارد.

مقایسه الگوریتم‌های Probabilistic و Resampling

در فرآیند انتخاب مدل (Model Selection)، الگوریتم‌های Probabilistic و Resampling هر دو نقش مهمی ایفا می‌کنند، اما از رویکردهای مختلفی برای دستیابی به هدف خود استفاده می‌کنند. این تفاوت‌ها در نحوه استفاده از داده‌ها، محاسبات و نتایج حاصل از آن‌ها قابل مشاهده است.

نحوه استفاده از داده‌ها:

الگوریتم‌های Probabilistic از تئوری احتمالات برای انتخاب مدل استفاده می‌کنند. این روش‌ها معمولاً شامل تخمین احتمال مدل‌های مختلف براساس داده‌های موجود و انتخاب مدلی با بیشترین احتمال یا بهترین معیار احتمال هستند. معیارهایی مانند Akaike Information Criterion (AIC) و Bayesian Information Criterion (BIC) از پیچیدگی مدل و کیفیت تناسب مدل با داده‌ها برای انتخاب بهترین مدل استفاده می‌کنند. به عبارتی، این روش‌ها به داده‌ها به عنوان یک کل نگاه می‌کنند و تلاش می‌کنند با استفاده از اطلاعات آماری، مدلی را پیدا کنند که تعادل مناسبی بین دقت و پیچیدگی داشته باشد.

در مقابل، الگوریتم‌های Resampling از تکنیک‌های بازنمونه‌گیری داده‌ها برای ارزیابی و انتخاب مدل استفاده می‌کنند. این روش‌ها شامل تقسیم داده‌ها به مجموعه‌های مختلف و انجام مکرر آموزش و ارزیابی مدل‌ها بر روی این مجموعه‌ها هستند. به عنوان مثال، در k-fold Cross-Validation، داده‌ها به k بخش تقسیم می‌شوند و مدل به صورت متوالی بر روی ترکیبات مختلف این بخش‌ها آموزش و ارزیابی می‌شود. روش Bootstrap نیز با نمونه‌گیری مکرر با بازگشت از داده‌ها، عملکرد مدل‌ها را بررسی می‌کند. این رویکردها به داده‌ها به عنوان مجموعه‌های متعدد و مجزا نگاه می‌کنند و با انجام مکرر آزمایشات، عملکرد مدل را تخمین می‌زنند.

پیچیدگی محاسباتی:

الگوریتم‌های Probabilistic نیاز به محاسبات پیچیده و زمان‌بر دارند. این روش‌ها به دلیل استفاده از تئوری‌های آماری و محاسبات احتمالاتی برای تخمین احتمال مدل‌ها و معیارهای اطلاعاتی، نیاز به پردازش

و تحلیل دقیق داده‌ها دارند. به همین دلیل، این روش‌ها ممکن است برای مجموعه داده‌های بسیار بزرگ یا مدل‌های بسیار پیچیده، زمان‌بر و محاسباتی سنگین باشند. همچنین، این الگوریتم‌ها نیاز به فرضیات قوی درباره توزیع داده‌ها و مدل‌ها دارند که ممکن است همیشه صحیح نباشند.

از سوی دیگر، الگوریتم‌های Resampling مانند Cross-Validation و Bootstrap، به دلیل بازنمونه‌گیری مکرر از داده‌ها، محاسبات نسبتاً ساده‌تری دارند. این روش‌ها با تقسیم داده‌ها به مجموعه‌های مختلف و انجام مکرر آزمایشات، عملکرد مدل‌ها را تخمین می‌زنند. با این حال، این روش‌ها نیز می‌توانند زمان‌بر باشند، به ویژه در مواردی که تعداد تکرارها یا بازنمونه‌گیری‌ها زیاد است. با این وجود، روش‌های Resampling معمولاً نیاز به فرضیات کمتری درباره توزیع داده‌ها دارند و انعطاف‌پذیری بیشتری در مواجهه با داده‌های غیرمعمول یا پیچیده دارند.

دقت و تعمیم‌پذیری

الگوریتم‌های Probabilistic معمولاً دقت بالاتری دارند و به دلیل توجه به پیچیدگی مدل‌ها، تعمیم‌پذیری بهتری نیز ارائه می‌دهند. این روش‌ها با استفاده از معیارهایی مانند AIC و BIC، مدلی را انتخاب می‌کنند که بهترین توازن بین تطابق با داده‌ها و ساده بودن را حفظ کند. این به این معناست که مدل انتخاب شده نه تنها دقت بالایی دارد، بلکه از پیچیدگی غیرضروری نیز جلوگیری می‌کند و در نتیجه تعمیم‌پذیری بهتری در مواجهه با داده‌های جدید خواهد داشت.

در مقابل، الگوریتم‌های Resampling به دلیل استفاده از بازنمونه‌گیری مکرر، تخمین‌های پایدارتری از عملکرد مدل ارائه می‌دهند. این روش‌ها به دلیل انجام مکرر آزمایشات و ارزیابی مدل‌ها بر روی مجموعه‌های مختلف داده‌ها، می‌توانند تخمین‌های پایدارتری از عملکرد مدل به‌دست آورند. این پایداری در تخمین عملکرد مدل، باعث می‌شود که این روش‌ها در مواجهه با داده‌های جدید، عملکرد قابل اعتمادتری داشته باشند.

انتخاب بین الگوریتم‌های Probabilistic و Resampling بستگی به نیازهای خاص مسئله، محدودیت‌های زمانی و محاسباتی، و میزان دانش درباره توزیع داده‌ها دارد. الگوریتم‌های Probabilistic معمولاً در شرایطی که دقت بالا و توجه به پیچیدگی مدل اهمیت دارد، مناسب‌تر هستند. در حالی که الگوریتم‌های Resampling به دلیل سادگی، انعطاف‌پذیری و پایداری بیشتر در تخمین عملکرد مدل، معمولاً در بسیاری از کاربردهای عملی ترجیح داده می‌شوند. هر دو دسته الگوریتم‌ها نقش مهمی در انتخاب مدل ایفا می‌کنند و استفاده ترکیبی از آن‌ها نیز می‌تواند در برخی موارد نتایج بهتری به همراه داشته باشد.

۳-۲_ انتخاب مدل با تعداد دادگان کم

چالش‌های انتخاب مدل

هنگامی که تعداد دادگان کم باشد، انتخاب و ارزیابی مدل با چالش‌های متعددی مواجه می‌شود. برخی از این چالش‌ها عبارتند از:

کمبود داده برای آموزش و ارزیابی:

یکی از بزرگ‌ترین چالش‌ها کمبود داده برای تقسیم به مجموعه‌های آموزشی و آزمون است. این مسئله می‌تواند منجر به آموزش ناکافی مدل و ارزیابی غیرمعتبر شود.

تعمیم‌پذیری ضعیف:

مدل‌هایی که با داده‌های کم آموزش دیده‌اند، ممکن است توانایی تعمیم به داده‌های جدید و دیده‌نشده را نداشته باشند. این مدل‌ها معمولاً به راحتی دچار بیش‌برازش (Overfitting) می‌شوند، به طوری که عملکرد خوبی روی داده‌های آموزشی دارند ولی روی داده‌های جدید عملکرد ضعیفی نشان می‌دهند.

ارزیابی نامعتبر:

با داده‌های کم، ارزیابی مدل‌ها با استفاده از روش‌های معمول اعتبارسنجی (مانند تقسیم داده‌ها به مجموعه‌های آموزشی و آزمون) ممکن است نتایج قابل اعتمادی نداشته باشد، زیرا اندازه کوچک مجموعه آزمون ممکن است نماینده خوبی از کل داده‌ها نباشد.

پیچیدگی مدل:

انتخاب مدل‌های پیچیده‌تر با داده‌های کم می‌تواند مشکل‌ساز باشد، زیرا مدل‌های پیچیده نیاز به داده‌های بیشتری برای آموزش دارند تا بتوانند الگوهای پیچیده‌تر را شناسایی کنند.

راه‌حل‌ها برای انتخاب و ارزیابی مدل با تعداد دادگان کم

استفاده از اعتبارسنجی متقابل (Cross-Validation):

اعتبارسنجی متقابل به‌ویژه k-fold Cross-Validation، یکی از بهترین روش‌ها برای ارزیابی مدل با داده‌های کم است. در این روش، داده‌ها به k بخش مساوی تقسیم می‌شوند و مدل به‌طور متوالی بر روی k-1 بخش آموزش داده شده و با بخش باقی‌مانده ارزیابی می‌شود. این فرآیند k بار تکرار می‌شود و میانگین نتایج

به‌دست آمده به عنوان عملکرد نهایی مدل در نظر گرفته می‌شود. این روش به استفاده بهینه از کل داده‌ها برای آموزش و ارزیابی کمک می‌کند.

Bootstrap:

روش Bootstrap شامل نمونه‌گیری مکرر با بازگشت از داده‌ها است. این روش می‌تواند به تخمین پایدارتری از عملکرد مدل کمک کند و ارزیابی معتبرتری ارائه دهد.

مدل‌های ساده‌تر:

انتخاب مدل‌های ساده‌تر که نیاز به داده‌های کمتری برای آموزش دارند، می‌تواند از بیش‌برازش جلوگیری کند. مدل‌های خطی یا مدل‌هایی با تعداد پارامترهای کمتر معمولاً در این شرایط عملکرد بهتری دارند.

تکنیک‌های افزایش داده (Data Augmentation):

در برخی موارد، می‌توان از تکنیک‌های افزایش داده برای تولید نمونه‌های جدید از داده‌های موجود استفاده کرد. این تکنیک‌ها معمولاً در حوزه‌هایی مانند پردازش تصویر و پردازش متن مورد استفاده قرار می‌گیرند.

استفاده از دانش قبلی و مدل‌های پیش‌آموزش‌دیده:

اگر مدل‌های پیش‌آموزش‌دیده یا دانش قبلی در دسترس باشد، می‌توان از آن‌ها برای انتقال یادگیری و بهبود عملکرد مدل با داده‌های کم استفاده کرد. این روش به ویژه در حوزه‌هایی مانند بینایی کامپیوتری و پردازش زبان طبیعی مؤثر است.

تنظیم هایپرپارامترها با روش‌های خاص:

به جای انجام جستجوی شبکه‌ای (Grid Search) که ممکن است نیاز به داده‌های زیادی داشته باشد، می‌توان از روش‌های جستجوی تصادفی (Random Search) یا بهینه‌سازی بیزین (Bayesian Optimization) استفاده کرد که کارایی بیشتری با داده‌های کم دارند.

۳_ پاسخ سوال ۳

پاسخ اسکن‌شده پیوست شد.

۴_ پاسخ سوال ۴

شبکه‌های عصبی چند لایه برای تخمین پارامترهای توزیع مخلوط گوسی به گونه‌ای طراحی می‌شوند که خروجی‌های آن‌ها شامل پارامترهای کلیدی GMM باشد. این پارامترها عبارتند از:

وزن‌های ترکیب (α): که نشان‌دهنده نسبت هر مؤلفه گوسی در مخلوط کلی است.

میانگین‌ها (μ): که مکان میانگین هر مؤلفه گوسی را مشخص می‌کند.

واریانس‌ها (σ^2): که پراکندگی داده‌ها در اطراف میانگین هر مؤلفه گوسی را نشان می‌دهد.

۴-۱_ خروجی‌های شبکه

وزن‌های ترکیب (α):

وزن‌های ترکیب (α) نشان‌دهنده احتمال هر مؤلفه گوسی در ترکیب کلی هستند و باید مجموع این وزن‌ها برابر با ۱ باشد.

برای تولید این وزن‌ها، از تابع Softmax در لایه خروجی استفاده می‌شود.

تعداد نورون‌ها: اگر تعداد مؤلفه‌های گوسی K باشد، تعداد نورون‌های مرتبط با وزن‌های ترکیب نیز K خواهد بود. هر نورون یک وزن (α k) را تولید می‌کند.

میانگین‌ها (μ) :

میانگین‌ها مقادیر پیوسته‌ای هستند که مکان هر مؤلفه گوسی را مشخص می‌کنند. برای به دست آوردن این پارامترها، خروجی خطی شبکه به طور مستقیم استفاده می‌شود. تعداد نورون‌ها: برای هر مؤلفه گوسی یک میانگین نیاز است. بنابراین، اگر تعداد مؤلفه‌های گوسی K باشد و هر مؤلفه در یک فضای d-بعدی تعریف شود تعداد نورون‌های مرتبط با میانگین‌ها K×d خواهد بود. به عبارتی برای هر مؤلفه d نورون برای میانگین داریم.

تابع فعال‌سازی: بدون تابع فعال‌سازی خاص (خروجی خطی)

واریانس‌ها (σ2) یا ماتریس‌های کوواریانس

واریانس‌ها باید مقادیر مثبتی باشند. برای اطمینان از مثبت بودن این پارامترها، از توابع فعال‌سازی مانند ReLU یا Softplus استفاده می‌شود.

تعداد نورون‌ها: برای هر مؤلفه گوسی یک واریانس نیاز است. بنابراین، اگر تعداد مؤلفه‌های گوسی K باشد و هر مؤلفه در یک فضای d-بعدی تعریف شود، تعداد نورون‌های مرتبط با واریانس‌ها K×d خواهد بود. به عبارتی برای هر مؤلفه d نورون برای واریانس داریم. اگر ماتریس‌های کوواریانس کامل باشند، هر ماتریس کوواریانس شامل d×d عنصر است، بنابراین تعداد نورون‌های مرتبط با کوواریانس‌ها K×d×d خواهد بود.

پس شبکه عصبی چند لایه برای تخمین پارامترهای GMM به صورت زیر طراحی می‌شود:

ورودی شبکه: داده‌های مشاهده شده که قرار است مدل‌سازی شوند.

لایه‌های پنهان: لایه‌های با نورون‌های متراکم که ویژگی‌های غیرخطی داده‌ها را استخراج می‌کنند.

لایه خروجی: شامل سه بخش مجزا برای تولید پارامترهای GMM:

وزن‌های ترکیب (α): با تابع Softmax و K نورون.

میانگین‌ها (μ): با خروجی خطی و K×d نورون.

واریانس‌ها (σ^2): برای واریانس‌های قطری: با تابع Softplus یا ReLU و K×d نورون

برای ماتریس‌های کوواریانس کامل: با تابع Softplus یا ReLU و K×d×d نورون.

۲-۴_ توابع فعالسازی مناسب برای خروجی

برای انتخاب توابع فعالسازی مناسب برای خروجی‌های شبکه عصبی که پارامترهای توزیع مخلوط گوسی (GMM) را تولید می‌کنند، باید ویژگی‌های هر پارامتر و نیازهای آن‌ها را در نظر بگیریم. این پارامترها شامل وزن‌های ترکیب (α)، میانگین‌ها (μ) و واریانس‌ها (σ^2) یا ماتریس‌های کوواریانس (Σ) هستند.

وزن‌های ترکیب (α)

مقادیر قابل قبول: وزن‌های ترکیب باید بین ۰ و ۱ باشند و مجموع آن‌ها برابر با ۱ باشد.

تابع فعالسازی پیشنهادی: Softmax

تابع Softmax تضمین می‌کند که خروجی‌های تولید شده بین ۰ و ۱ هستند و مجموع آن‌ها برابر با ۱ خواهد بود. این تابع به طور معمول برای دسته‌بندی چند کلاسه استفاده می‌شود و در اینجا نیز برای تولید وزن‌های ترکیب مناسب است.

میانگین‌ها (μ)

مقادیر قابل قبول: میانگین‌ها می‌توانند هر مقداری باشند، زیرا مکان مرکز هر مؤلفه گوسی را مشخص می‌کنند.

تابع فعالسازی پیشنهادی: بدون تابع فعالسازی خاص (خروجی خطی)

برای تولید میانگین‌ها نیازی به تابع فعالسازی خاصی نیست زیرا این مقادیر می‌توانند هر عددی در فضای ویژگی‌ها باشند. بنابراین، خروجی خطی مستقیم برای این منظور مناسب است.

واریانس‌ها (σ^2) یا ماتریس‌های کوواریانس (Σ)

مقادیر قابل قبول: واریانس‌ها و عناصر ماتریس‌های کوواریانس باید مقادیر مثبتی باشند. در صورت استفاده از ماتریس کوواریانس کامل، ماتریس باید نیمه معین مثبت یا Positive Semi-definite باشد.

واریانس‌های قطری:

تابع فعالسازی پیشنهادی: ReLU یا Softplus یا Exponential

تابع Softplus: این تابع تضمین می‌کند که خروجی همیشه مثبت است.

تابع ReLU: این تابع نیز خروجی‌های منفی را به صفر نگاشت می‌کند و خروجی‌های مثبت را بدون تغییر می‌گذارد.

تابع Exponential: این تابع نیز خروجی‌های مثبت تولید می‌کند.

ماتریس‌های کوواریانس کامل:

تابع فعالسازی پیشنهادی:

عناصر پایین مثلثی: خروجی خطی یا بدون تابع فعالسازی خاص - عناصر قطری: Exponential

۳-۴_ تابع هزینه

تابع هزینه Negative Log-Likelihood یکی از رایج‌ترین معیارهای بهینه‌سازی در مدل‌های آماری و شبکه‌های عصبی است که برای تخمین پارامترهای توزیع مخلوط گوسی (GMM) استفاده می‌شود. این تابع هزینه میزان خطا در تخمین پارامترهای مدل را کاهش می‌دهد و به صورت زیر تعریف می‌شود:

$$L = -\sum_{i=1}^{N} \log(\sum_{k=1}^{K} \alpha_k \boldsymbol{N}(x_i|\mu_k, \Sigma_k))$$

L تابع هزینه است.

N تعداد نمونه‌های داده است.

K تعداد مؤلفه‌های گوسی است.

α k وزن ترکیب برای مؤلفه k ام است.

$\boldsymbol{N}(x_i|\mu_k, \Sigma_k)$ تابع چگالی گوسی با میانگین μ k و کوواریانس Σ k برای نمونه x i است.

بیشینه‌سازی احتمال:

احتمال کلی داده‌ها: هدف اصلی در مدل‌های آماری، بیشینه‌سازی احتمال مشاهده داده‌ها تحت مدل مشخص شده است. تابع احتمال (Likelihood Function) بیانگر احتمال مشاهده داده‌های واقعی تحت پارامترهای تخمین زده شده مدل است.

لگاریتم تابع احتمال: برای ساده‌سازی محاسبات، معمولاً از لگاریتم تابع احتمال استفاده می‌شود. لگاریتم تابع احتمال ویژگی‌های مفیدی دارد که محاسبات را تسهیل می‌کند، از جمله تبدیل ضرب به جمع و جلوگیری از وقوع اعداد بسیار کوچک.

منفی لگاریتم احتمال:

مقدار منفی: چون هدف، بیشینه‌سازی احتمال است، باید تابعی تعریف شود که در صورت بهینه بودن پارامترها مقدار کمتری داشته باشد. به همین دلیل از منفی لگاریتم احتمال استفاده می‌شود که باید مینیمم شود.

تابع محدب: لگاریتم احتمال در بسیاری از موارد تابعی محدب است که بهینه‌سازی آن را ساده‌تر می‌کند و الگوریتم‌های بهینه‌سازی می‌توانند به راحتی به نقطه بهینه همگرا شوند.

۵_ پاسخ سوال ۵

ابتدا کتابخانه‌های مورد نیاز را Import می‌کنیم.

۵-۱_ بارگذاری داده‌ها و خروجی تصویر دلخواه

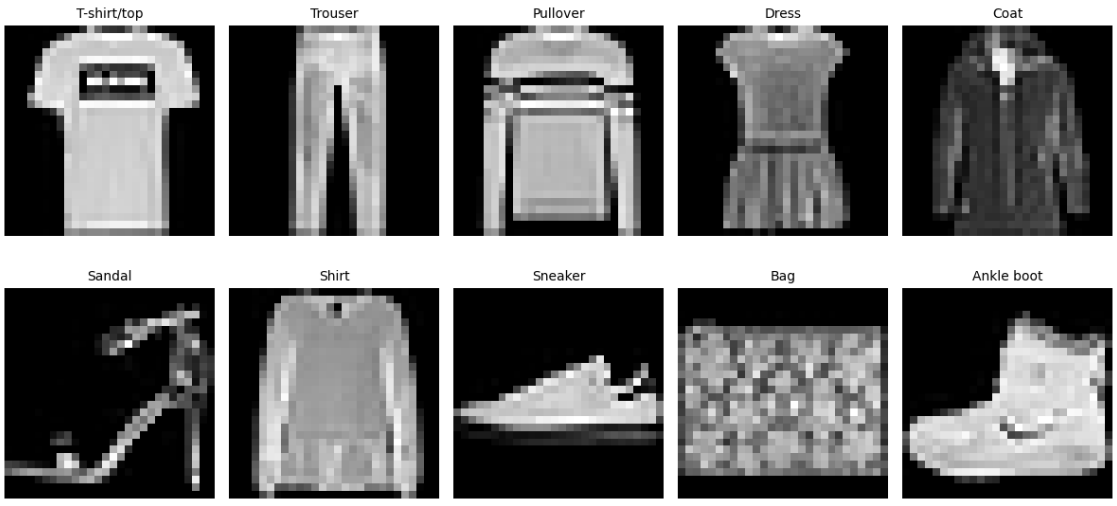
```
mnist = fetch_openml('Fashion-MNIST', version=1)
X = mnist.data
y = mnist.target.astype(int)
```

در این قسمت از کد، ابتدا مجموعه داده‌ی Fashion-MNIST را با استفاده از تابع fetch_openml از کتابخانه‌ی sklearn.datasets بارگذاری می‌کنیم. این داده شامل تصاویر مختلف از پوشاک (کفش، تی‌شرت و غیره) است که به صورت عددی ذخیره شده‌اند.

متغیر X حاوی داده‌های تصویری است که هر تصویر به صورت یک بردار با طول ۷۸۴ (۲۸در ۲۸ پیکسل) نمایش داده شده است. متغیر y نیز برچسب‌های مرتبط با هر تصویر را نگهداری می‌کند که نشان‌دهنده‌ی نوع پوشاک موجود در تصویر است. با استفاده از astype(int) برچسب‌ها را به نوع عدد صحیح تبدیل می‌کنیم تا بتوانیم به راحتی از آن‌ها در مراحل بعدی استفاده کنیم.

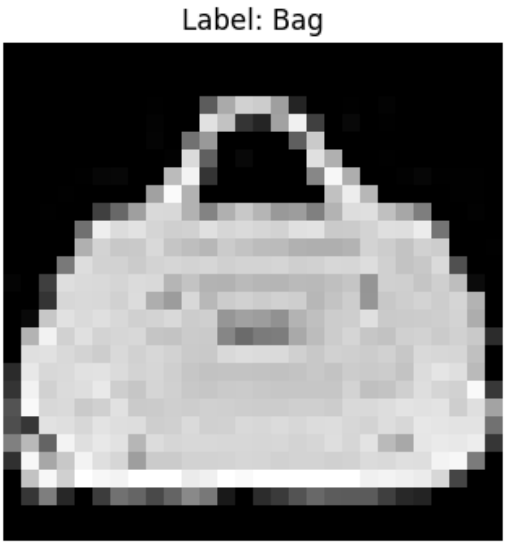
ابتدا شکل مجموعه داده‌ها را نمایش می‌دهیم تا اطلاعات اولیه درباره اندازه داده‌ها و تعداد ویژگی‌ها کسب کنیم.

سپس با استفاده از np.unique برچسب‌های یکتا را استخراج کرده و برای هر برچسب، اولین تصویر مرتبط با آن را پیدا کرده و برای هر برچسب یک تصویر نمایش می‌دهیم. شکل زیر خروجی این قسمت از کد است:



شکل ۱ تصاویری از تمام برچسب‌های Fashion MNIST

سپس یک تصویر تصادفی از مجموعه داده انتخاب کرده و سپس آن را رسم می‌کنیم. ابتدا یک اندیس تصادفی بین صفر تا تعداد کل تصاویر موجود در مجموعه داده انتخاب می‌کنیم. سپس تصویر متناظر با این اندیس را از آرایه داده‌ها استخراج کرده و شکل آن را به ۲۸ در ۲۸ Reshape می‌کنیم تا به صورت تصویری قابل مشاهده باشد. برچسب مربوط به این تصویر نیز از آرایه برچسب‌ها استخراج می‌کنیم. این آرایه به منظور نمایش برچسب داده‌ها پیاده‌سازی شد.



شکل ۲ یکی از تصاویر مجموعه Fashion MNIST

۵-۲_ استانداردسازی داده‌ها

در این بخش از کد، داده‌های ویژگی‌ها (X) با استفاده از روش استانداردسازی، نرمال می‌شوند. استانداردسازی به این معناست که هر ویژگی داده‌ها به گونه‌ای تغییر می‌کند که میانگین آن برابر با صفر و واریانس آن برابر با یک شود. این کار باعث می‌شود که ویژگی‌ها با مقیاس‌های مختلف، تأثیر یکسانی در مدل‌سازی داشته باشند.

```
scaler = StandardScaler()
X_standardized = scaler.fit_transform(X_array)
```

ابتدا از StandardScaler از کتابخانه sklearn.preprocessing استفاده می‌کنیم. سپس با استفاده از متد fit_transform، داده‌های تصویری استانداردسازی می‌شوند.

۵-۳_ ماتریس کواریانس و نمایش ابعاد آن

ماتریس کواریانس به ما نشان می‌دهد که چگونه ویژگی‌های مختلف داده‌ها با هم تغییر می‌کنند و همبستگی بین آن‌ها چگونه است. ابتدا، با استفاده از تابع np.cov از کتابخانه numpy، ماتریس کواریانس را محاسبه می‌کنیم. برای این کار، داده‌های استانداردسازی شده را ترانهاده (transpose) می‌کنیم تا هر ستون نماینده یک ویژگی باشد.

به دلیل اینکه داده‌های اصلی ما دارای ۷۸۴ ویژگی (پیکسل) هستند، ماتریس کواریانس نیز ابعادی برابر با ۷۸۴ در ۷۸۴ خواهد داشت. این ماتریس تمامی جفت‌های ممکن از ویژگی‌ها و همبستگی بین آن‌ها را شامل می‌شود.

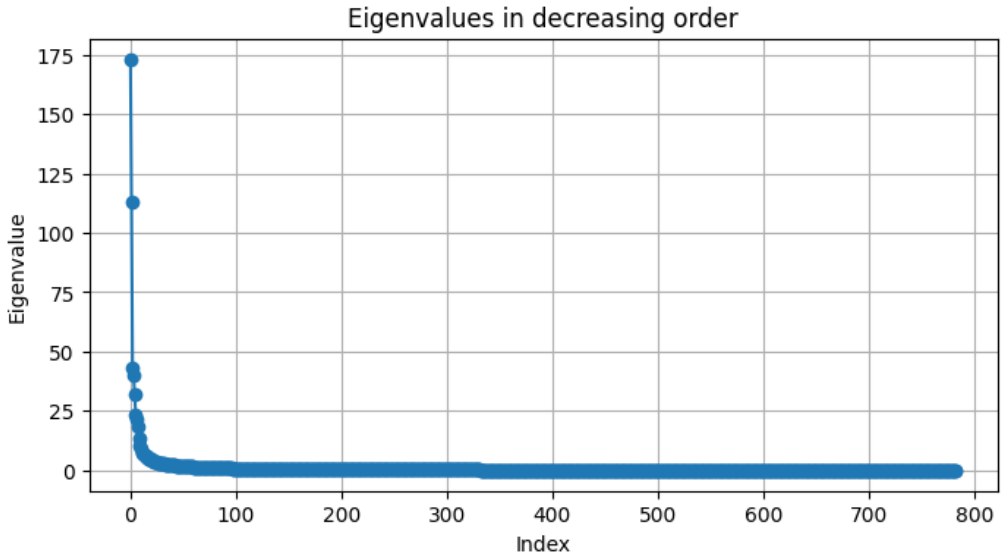
Covariance matrix shape: (784, 784)

۴-۵_ محاسبه و رسم مقادیر ویژه و بردارهای ویژه

در این بخش، ابتدا مقادیر ویژه و بردارهای ویژه ماتریس کواریانس را محاسبه می‌کنیم. مقادیر ویژه نشان‌دهنده میزان واریانسی است که هر مؤلفه اصلی (Principal Component) در داده‌ها توضیح می‌دهد. بردارهای ویژه نیز جهت مؤلفه‌های اصلی را تعیین می‌کنند. ابتدا، با استفاده از تابع np.linalg.eigh از کتابخانه numpy، مقادیر ویژه و بردارهای ویژه ماتریس کواریانس را محاسبه می‌کنیم. سپس، مقادیر ویژه را به ترتیب کاهشی مرتب کرده و بردارهای ویژه متناظر با آن‌ها را نیز ذخیره می‌کنیم.

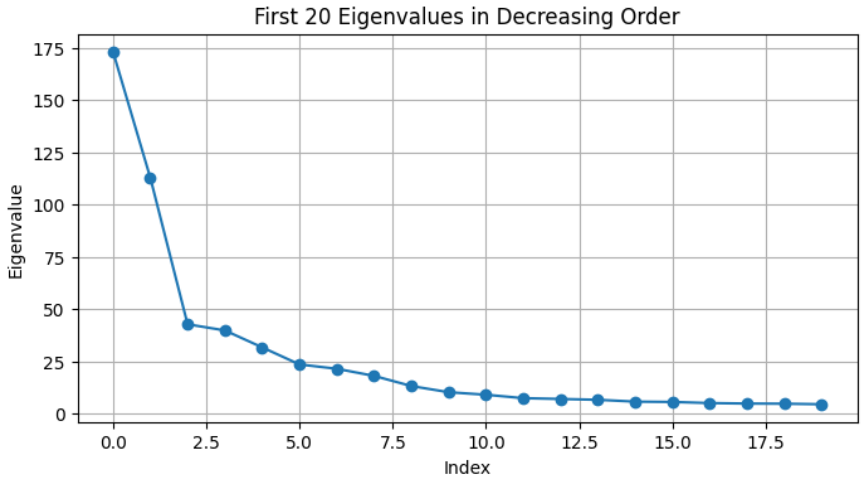
```
eigenvalues, eigenvectors = np.linalg.eigh(cov_matrix)
sorted_indices = np.argsort(eigenvalues)[::-1]
eigenvalues_sorted = eigenvalues[sorted_indices]
eigenvectors_sorted = eigenvectors[:, sorted_indices]
```

در مرحله بعد، مقادیر ویژه مرتب‌شده را رسم می‌کنیم تا ببینیم چگونه این مقادیر کاهش می‌یابند.



شکل ۳ مقادیر ویژه به ترتیب نزولی

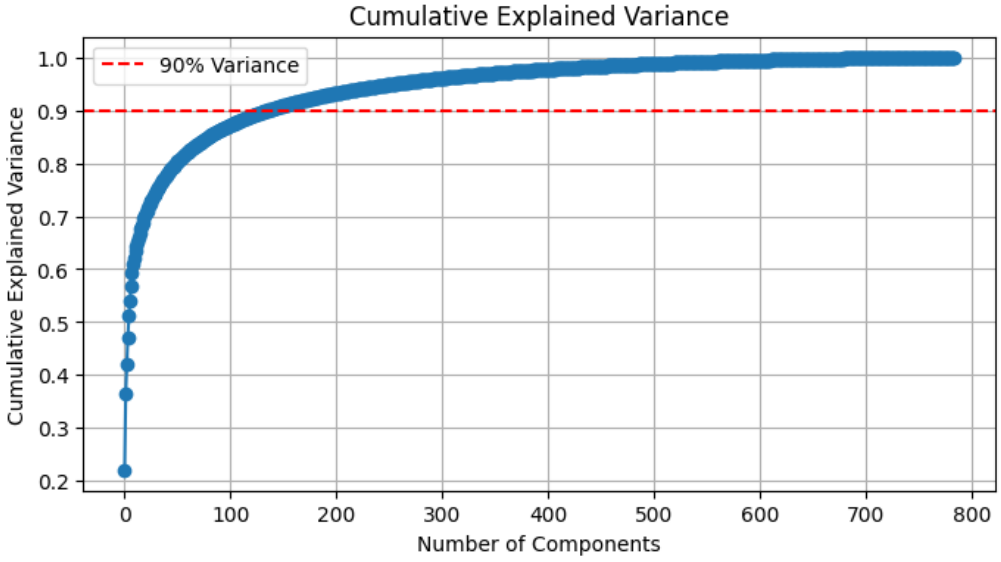
برای مشاهده بهتر نمودار مربوط به ۲۰ مقدار اول نیز خروجی گرفته شد:



شکل ۴ مقادیر ویژه به ترتیب نزولی – ۲۰ مورد اول

تعیین تعداد کامپوننت‌های مناسب

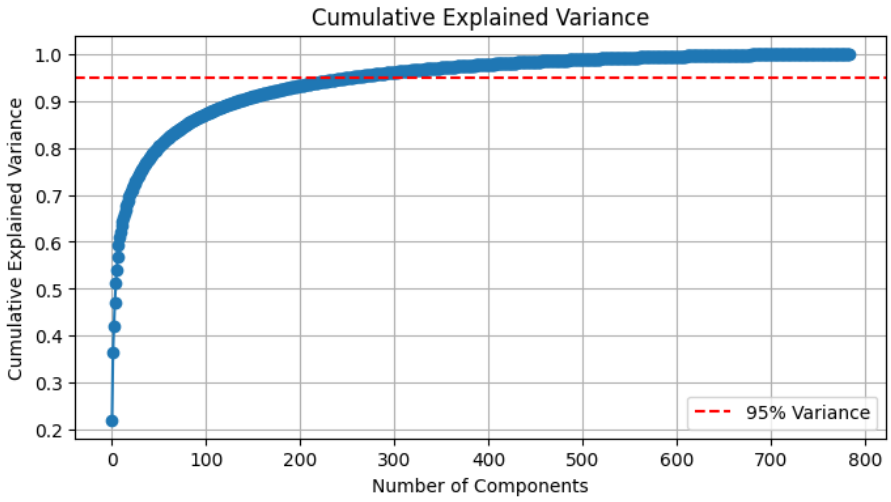
برای تعیین تعداد کامپوننت‌های مناسب، نسبت واریانس توضیح داده شده توسط هر مؤلفه را محاسبه کرده و سپس واریانس تجمعی را رسم می‌کنیم. با این کار می‌توانیم تعداد مؤلفه‌هایی که برای حفظ درصد مشخصی از واریانس (مانند ۹۰٪ یا ۹۵٪) لازم است را تعیین کنیم. در منابع اینترنتی گفته شد ۹۵ درصد ولی در اسلایدهای درسی ۹۰ درصد بود. در این تمرین ما هردوی آن‌ها را پیدا و رسم کردیم.



شکل ۵ واریانس تجمعی – ۹۰ درصد

برای انتخاب تعداد کامپوننت‌های مناسب در فرآیند فشرده‌سازی، معمولاً به دنبال حفظ درصد بالایی از واریانس کل داده‌ها هستیم. درصدهایی مانند ۹۰٪ و ۹۵٪ از واریانس کل به عنوان معیارهای رایج در نظر گرفته می‌شوند. با رسم واریانس تجمعی، می‌توانیم ببینیم که چند مؤلفه اول چند درصد از واریانس را توضیح

می‌دهند. سپس با توجه به نمودار، تعداد کامپوننت‌هایی که برای حفظ این درصد از واریانس کافی هستند را انتخاب می‌کنیم.



شکل ۶ واریانس تجمعی – ۹۵ درصد

در این مثال، برای حفظ ۹۰٪ از واریانس داده‌ها، تعداد **۱۳۷** کامپوننت و برای حفظ ۹۵٪ از واریانس، تعداد **۲۵۶** کامپوننت کافی است. این تعداد کامپوننت‌ها به ما کمک می‌کنند تا با کاهش ابعاد داده‌ها، همچنان اطلاعات اصلی و مهم داده‌ها را حفظ کنیم و مدل‌های یادگیری ماشین را به صورت کارآمدتری پیاده‌سازی کنیم.

۵-۵_ فشرده‌سازی و بازسازی تصاویر

برای این کار، ابتدا تعداد کامپوننت‌های مناسب برای حفظ درصد مشخصی از واریانس داده‌ها (در اینجا ۹۵٪) را انتخاب کرده و سپس فرآیند فشرده‌سازی و بازسازی را انجام می‌دهیم. با استفاده از بردارهای ویژه مربوط به ۲۵۶ کامپوننت اول، داده‌های استاندارد شده را فشرده می‌کنیم.

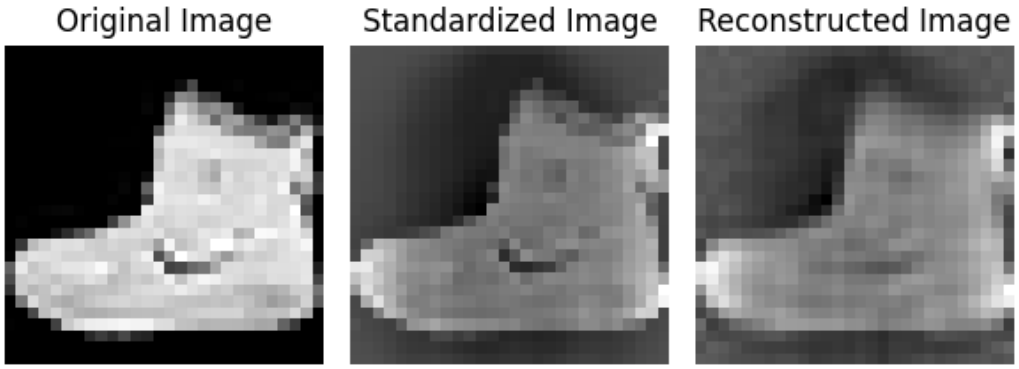
```
n_components = 256
selected_eigenvectors = eigenvectors_sorted[:, :n_components]
```

سپس داده‌های فشرده شده را با استفاده از همان بردارهای ویژه بازسازی می‌کنیم. در این پروژه، به جای استفاده از توابع آماده کتابخانه‌های موجود، فرآیند تحلیل مؤلفه‌های اصلی (PCA) را به صورت دستی پیاده‌سازی کردیم.

```
X_pca = np.dot(X_standardized, selected_eigenvectors)
```

۶-۵_ تحلیل نتایج

در این بخش، سه تصویر شامل تصویر اصلی، تصویر استاندارد شده و تصویر بازسازی شده را مشاهده می‌کنیم. هدف این است که ببینیم تا چه حد توانسته‌ایم اطلاعات اصلی تصویر را پس از فشرده‌سازی و بازسازی حفظ کنیم.



شکل ۷ خروجی تصاویر مراحل مختلف

تصویر اصلی (Original Image):

این تصویر نمایانگر داده‌های اولیه است که هیچگونه تغییر یا پیش‌پردازشی بر روی آن اعمال نشده است. این تصویر نشان‌دهنده تمام جزئیات و اطلاعات اصلی موجود در داده اولیه است.

تصویر استاندارد شده (Standardized Image):

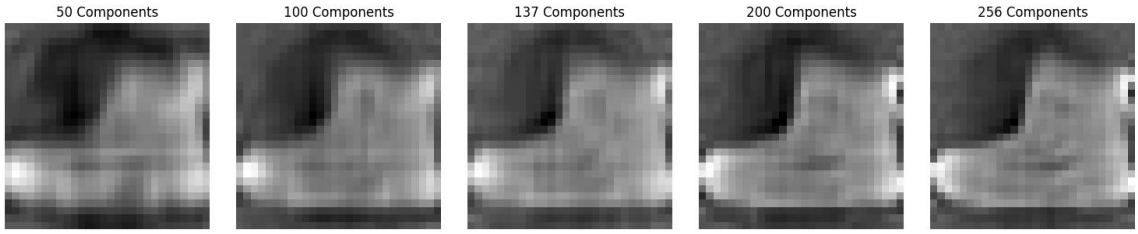
تصویر استاندارد شده پس از اعمال فرآیند استانداردسازی بر روی داده‌ها به دست آمده است. در این مرحله، میانگین هر ویژگی به صفر و واریانس آن به یک تغییر یافته است. این تصویر ممکن است تفاوت‌هایی در روشنایی و کنتراست نسبت به تصویر اصلی داشته باشد، اما همچنان ساختار کلی تصویر حفظ شده است.

تصویر بازسازی شده (Reconstructed Image):

این تصویر پس از فشرده‌سازی داده‌ها با استفاده از PCA و سپس بازسازی آن‌ها به دست آمده است. برای فشرده‌سازی، ۲۵۶ مؤلفه اصلی انتخاب شده‌اند که ۹۵٪ از واریانس کل داده‌ها را حفظ می‌کنند. تصویر بازسازی شده به دلیل فشرده‌سازی ممکن است برخی از جزئیات خود را از دست داده باشد و کیفیت آن نسبت به تصویر اصلی کاهش یافته است. با این حال، ساختار کلی و شکل اصلی تصویر همچنان قابل تشخیص است.

این نتیجه نشان می‌دهد که PCA می‌تواند به طور مؤثری ابعاد داده‌ها را کاهش دهد و در عین حال بخش اعظم اطلاعات مهم را حفظ کند. این ویژگی باعث می‌شود که PCA یک ابزار قدرتمند برای کاهش ابعاد داده‌های بزرگ و پیچیده باشد.

۷-۵_ بازسازی تصاویر با تعداد کامپوننت‌های مختلف



شکل ۸ نتایج بازسازی تصاویر با تعداد کامپوننت‌های مختلف

تعداد کمتر از ۱۰۰ کامپوننت: کیفیت تصاویر بازسازی شده با تعداد کمتر از ۱۰۰ کامپوننت پایین است و بسیاری از جزئیات از دست می‌روند. این نشان می‌دهد که این تعداد کامپوننت برای حفظ اطلاعات کافی نیست.

۱۳۷ کامپوننت: با استفاده از ۱۳۷ کامپوننت، کیفیت تصویر بازسازی شده بهبود یافته و جزئیات بیشتری حفظ شده‌اند. این تعداد کامپوننت برای حفظ ۹۰٪ از واریانس داده‌ها کافی است و نتیجه قابل قبولی ارائه می‌دهد.

۲۰۰ و ۲۵۶ کامپوننت: تصاویر بازسازی شده با ۲۰۰ و ۲۵۶ کامپوننت کیفیت بسیار بالایی دارند و به تصویر اصلی نزدیک‌تر هستند. این نشان می‌دهد که با افزایش تعداد کامپوننت‌ها، کیفیت تصویر بازسازی شده نیز بهبود می‌یابد.

۶_ پاسخ سوال ۶

۶-۱_ کاهش ابعاد با استفاده از PCA و برازش تابع مخلوط گوسی

بارگذاری داده‌ها و انتخاب کلاس‌های مورد نظر

ابتدا مجموعه داده MNIST را بارگذاری کرده و فقط نمونه‌های مربوط به کلاس‌های ۰ و ۱ را انتخاب می‌کنیم. در این قسمت، از کتابخانه fetch_openml برای بارگذاری داده‌های MNIST استفاده می‌کنیم. سپس با استفاده از ماسک، فقط نمونه‌های مربوط به اعداد ۰ و ۱ را فیلتر کرده و به آرایه تبدیل می‌کنیم.

```
mnist = fetch_openml('mnist_784', version=1, parser='auto')
X = mnist.data
y = mnist.target.astype(int)

mask = (y == 0) | (y == 1)
X = X[mask]
y = y[mask]

X_array = X.to_numpy()
```

```
Shape of X_array: (14780, 784)
```

هر تصویر به صورت یک بردار با طول ۷۸۴ (۲۸در ۲۸ پیکسل) نمایش داده شده است.

کاهش ابعاد با استفاده از PCA

در این مرحله، با استفاده از PCA، ابعاد داده‌ها را از ۷۸۴ به ۲ کاهش می‌دهیم.

```
pca = PCA(n_components=2)
X_reduced = pca.fit_transform(X_array)
```

برازش تابع مخلوط گوسی (GMM)

تابع مخلوط گوسی با دو جز (دو مؤلفه) را بر روی داده‌های کاهش‌یافته برازش می‌کنیم. GMM یک مدل احتمالاتی است که فرض می‌کند داده‌ها از ترکیب چند توزیع گوسی به دست آمده‌اند.

```
gmm = GaussianMixture(n_components=2)
gmm.fit(X_reduced)
```

در اینجا، یک شیء از کلاس GaussianMixture با تعداد مؤلفه‌های ۲ ایجاد می‌کنیم و سپس با استفاده از متد fit داده‌های کاهش‌یافته را بر روی مدل برازش می‌دهیم.

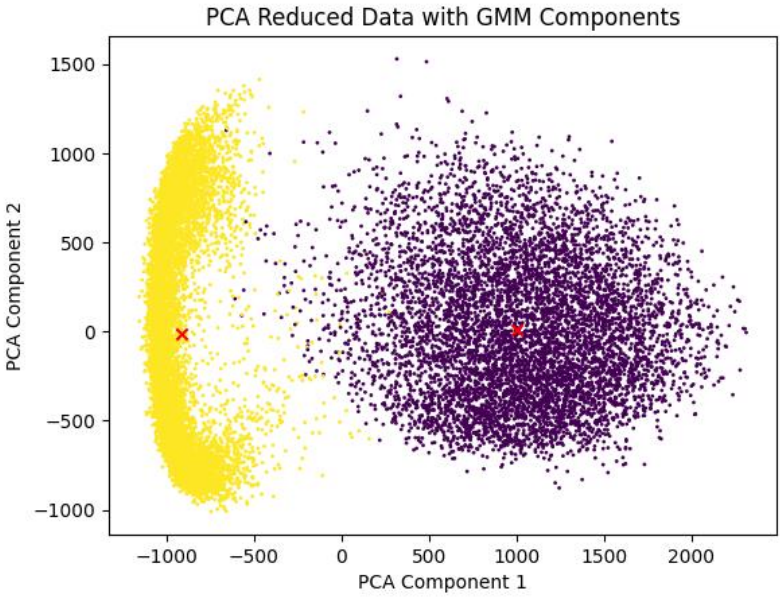
رسم داده‌ها و جزهای GMM

مقادیر میانگین و کوواریانس‌های هر جز محاسبه شده و نمایش می‌دهیم:

```
Means:
 [[1006.19031157   10.51864427]
```

```
 [-909.37840535   -9.50657926]]
Covariances:
 [[[238645.12579746 -13106.69055459]
  [-13106.69055459 155480.85587441]]

 [[ 15765.73998177  -6364.89252409]
  [ -6364.89252409 412967.18912251]]]
```



نمودار نشان می‌دهد که داده‌های کاهش‌یافته به دو خوشه جداگانه تقسیم شده‌اند و مراکز این خوشه‌ها با علامت‌های قرمز "x" نشان داده شده‌اند.

۲-۶_ اختلاف بین مقادیر میانگین هر کدام از دو جز تابع مخلوط گوسی

فاصله اقلیدسی یک معیار ساده و موثر برای اندازه‌گیری فاصله بین دو نقطه در فضای چندبعدی است.

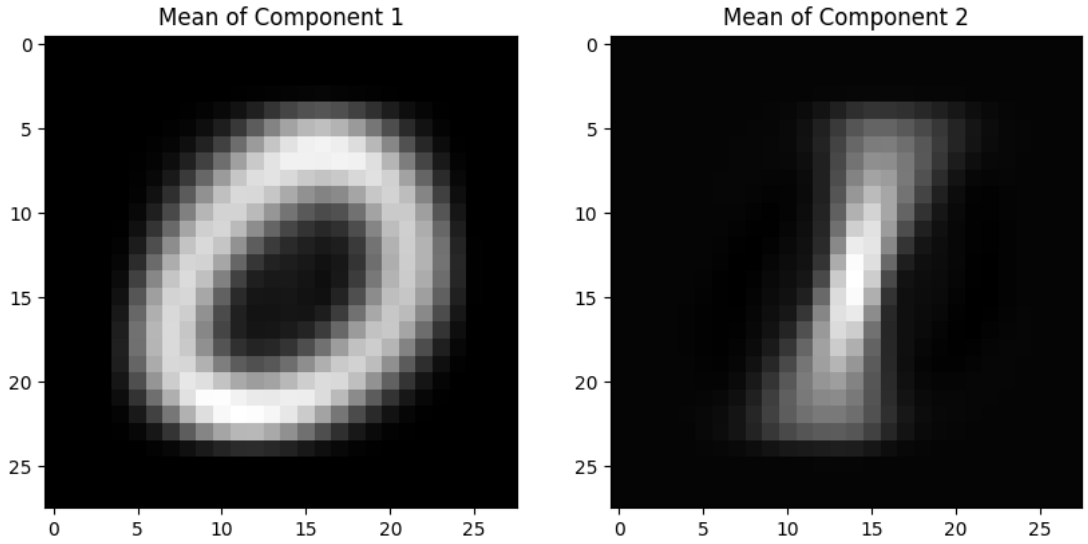
```
distance = euclidean(means[0], means[1])
```

فاصله اقلیدسی بین مقادیر میانگین دو جز تابع مخلوط گوسی برابر با ۱۹۱۵,۶۷ است. این مقدار نشان‌دهنده فاصله نسبتاً زیادی بین مراکز دو خوشه در فضای کاهش‌یافته دو بعدی است. فاصله بزرگ بین این دو میانگین نشان می‌دهد که دو جز تابع مخلوط گوسی به خوبی از هم تفکیک شده‌اند و داده‌های دو کلاس (اعداد ۰ و ۱) به طور موثر به دو خوشه جداگانه تقسیم شده‌اند.

این نتیجه بیانگر موفقیت PCA در کاهش ابعاد داده‌ها به دو بعد و GMM در خوشه‌بندی داده‌های کاهش‌یافته است. به عبارت دیگر، این روش‌ها توانسته‌اند ساختار داده‌ها را به خوبی حفظ کنند و دو کلاس مختلف را از هم جدا کنند.

بازگشت به فضای اصلی و نمایش مقادیر میانگین جزهای تابع مخلوط گوسی

با استفاده از معکوس PCA، مقادیر میانگین هر کدام از جزهای تابع مخلوط گوسی را از فضای دو بعدی به فضای ۷۸۴ بعدی برمی‌گردانیم. تصاویر بازسازی‌شده مربوط به هر کدام از جزهای تابع مخلوط گوسی را به صورت تصویر نمایش می‌دهیم.



شکل ۹ تصاویر بازسازی‌شده مربوط به هر کدام از جزهای تابع مخلوط گوسی

تصویر مربوط به میانگین جز اول نشان‌دهنده یک دایره (عدد ۰) است. این تصویر واضح است و به خوبی شکل عدد ۰ را نمایش می‌دهد. این نشان می‌دهد که جز اول تابع مخلوط گوسی عمدتاً نماینده داده‌های کلاس عدد ۰ است.

تصویر مربوط به میانگین جز دوم نمایانگر یک خط عمودی (عدد ۱) است. این تصویر نیز واضح است و به خوبی شکل عدد ۱ را نمایش می‌دهد. این نشان می‌دهد که جز دوم تابع مخلوط گوسی عمدتاً نماینده داده‌های کلاس عدد ۱ است. بازگرداندن مقادیر میانگین جزهای تابع مخلوط گوسی به فضای اصلی با استفاده از عکس PCA و نمایش آن‌ها به صورت تصاویر، نشان می‌دهد که دو جز تابع مخلوط گوسی به خوبی نمایانگر دو کلاس مختلف از داده‌های MNIST (اعداد ۰ و ۱) هستند.

۳-۶_ نمونه‌هایی با کمترین تفاوت در احتمال تعلق به هر دو جز

یافتن نمونه‌هایی با کمترین تفاوت در احتمال تعلق به هر دو جز تابع مخلوط گوسی نشان می‌دهد که این نمونه‌ها دارای ویژگی‌های مشترکی بین دو کلاس هستند. تصاویر بازگردانده شده به فضای اصلی این واقعیت را نشان می‌دهند که این نمونه‌ها ترکیبی از ویژگی‌های دو کلاس مختلف را دارند

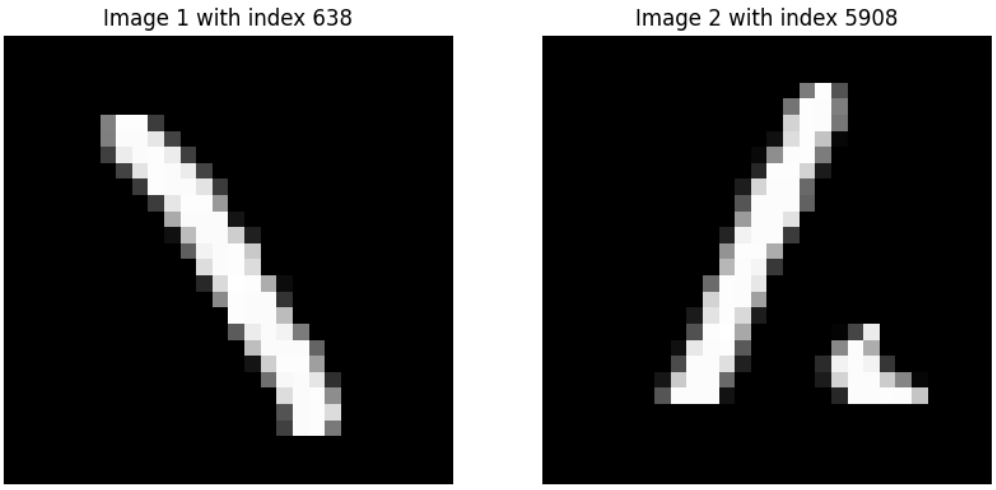
ابتدا احتمال تعلق هر نمونه به هر دو جز تابع مخلوط گوسی را محاسبه می‌کنیم.

```
probs = gmm.predict_proba(X_reduced)
prob_differences = np.abs(probs[:, 0] - probs[:, 1])

min_diff_indices = np.argsort(prob_differences)[:2]
```

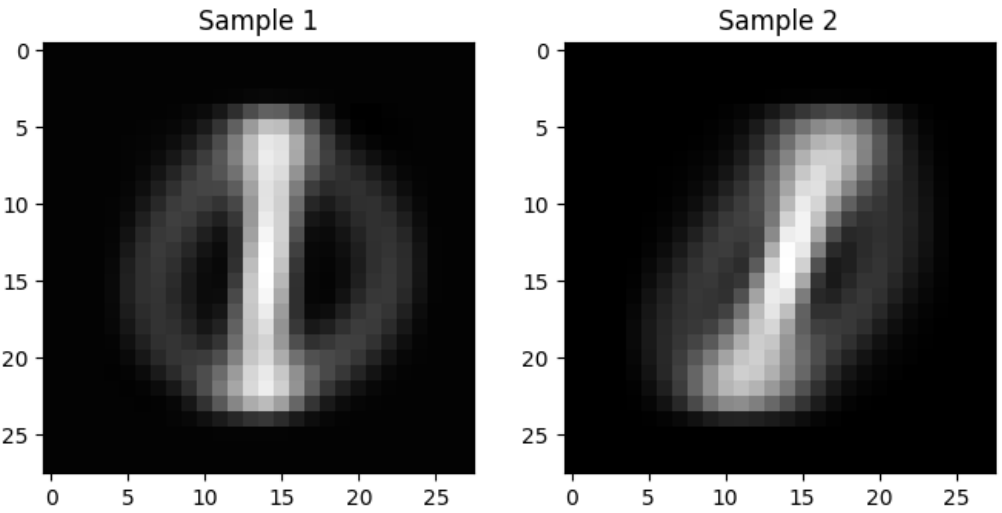
اختلاف مطلق بین این احتمالات را محاسبه کرده و دو نمونه با کمترین اختلاف را پیدا می‌کنیم.

تصاویر اصلی این نمونه‌ها را در فضای ۷۸۴ بعدی :



شکل ۱۰ تصاویر اصلی

نمونه‌های کاهش‌یافته را با استفاده از عکس PCA به فضای اصلی برمی‌گردانیم و به صورت تصویر نمایش می‌دهیم.



Sample 1: تصویر بازگردانده شده به فضای اصلی که ترکیبی از ویژگی‌های اعداد ۰ و ۱ را نشان می‌دهد. این تصویر شبیه یک عدد ۰ است که درون آن خطی شبیه عدد ۱ وجود دارد.

Sample 2: تصویر بازگردانده شده به فضای اصلی که بیشتر شبیه عدد ۱ است اما کمی ویژگی‌های عدد ۰ را نیز دارد.

۴-۶_ اختلاف میانگین‌های تابع مخلوط گوسی برای جفت کلاس‌های غیرهمسان

اختلاف بین میانگین‌های دو جز تابع مخلوط گوسی نشان‌دهنده تفاوت یا شباهت بین داده‌های دو کلاس است. جفت کلاس‌هایی که بیشترین اختلاف را دارند، به طور قابل توجهی از هم متمایز هستند و داده‌های آن‌ها به خوبی از یکدیگر جدا می‌شوند. در مقابل، جفت کلاس‌هایی که کمترین اختلاف را دارند، احتمالاً دارای شباهت‌های بیشتری هستند و داده‌های آن‌ها ممکن است همپوشانی بیشتری داشته باشند.

در کد زیر برای هر جفت کلاس غیرهمسان، داده‌های مربوط به این دو کلاس را انتخاب می‌کنیم، سپس با استفاده از PCA ابعاد داده‌ها را به ۲ کاهش می‌دهیم و GMM با دو جز را بر روی داده‌های کاهش‌یافته برازش می‌دهیم. اختلاف بین میانگین‌های دو جز تابع مخلوط گوسی با استفاده از فاصله اقلیدسی محاسبه شده و جفت کلاس‌هایی که بیشترین و کمترین اختلاف بین میانگین‌هایشان وجود دارد، شناسایی و ذخیره می‌کنیم.

```
X = mnist.data
y = mnist.target.astype(int)

max_diff = -np.inf
min_diff = np.inf
max_pair = None
min_pair = None


for (class1, class2) in combinations(range(10), 2):
    mask = (y == class1) | (y == class2)
    X_pair = X[mask]
    y_pair = y[mask]


    X_pair_array = X_pair.to_numpy()


    pca = PCA(n_components=2)
    X_reduced = pca.fit_transform(X_pair_array)


    gmm = GaussianMixture(n_components=2)
    gmm.fit(X_reduced)


    means = gmm.means_


    distance = euclidean(means[0], means[1])


    if distance > max_diff:
        max_diff = distance
        max_pair = (class1, class2)


    if distance < min_diff:
        min_diff = distance
        min_pair = (class1, class2)
```

۵-۶_ نتیجه‌گیری

بیشترین اختلاف بین میانگین‌ها:

جفت کلاس‌ها: (۰, ۱)

فاصله اقلیدسی بین میانگین‌ها: **۱۹۱۵,۶۷**

کمترین اختلاف بین میانگین‌ها:

جفت کلاس‌ها: (**۸, ۹**)

فاصله اقلیدسی بین میانگین‌ها: **۹۲۷,۴۸**

کلاس‌های با بیشترین اختلاف (۰, ۱):

جفت کلاس‌های (۰, ۱) که بیشترین اختلاف بین میانگین‌هایشان را دارند، نشان می‌دهند که این دو کلاس به خوبی از یکدیگر متمایز هستند. عدد ۰ و عدد ۱ دارای ویژگی‌های تصویری بسیار متفاوتی هستند که باعث می‌شود داده‌های آن‌ها به خوبی از هم جدا شوند. به عنوان مثال، عدد ۰ به صورت دایره‌ای است در حالی که عدد ۱ به صورت یک خط عمودی است. این تفاوت‌های ساختاری باعث می‌شود که میانگین‌های این دو کلاس در فضای کاهش‌یافته با PCA به طور قابل توجهی از هم فاصله بگیرند.

جفت کلاس‌های با کمترین اختلاف (۸, ۹):

جفت کلاس‌های (۸, ۹) که کمترین اختلاف بین میانگین‌هایشان را دارند، نشان می‌دهند که این دو کلاس دارای شباهت‌های بیشتری هستند و داده‌های آن‌ها ممکن است همپوشانی بیشتری داشته باشند. عدد ۸ و عدد ۹ دارای ساختارهای تصویری مشابهی هستند؛ به عنوان مثال، هر دو عدد دارای حلقه‌ها و خطوط خمیده‌ای هستند که باعث می‌شود ویژگی‌های تصویری آن‌ها شبیه به هم باشد. این شباهت‌ها باعث می‌شود که میانگین‌های این دو کلاس در فضای کاهش‌یافته با PCA به هم نزدیک‌تر باشند.

۷_ پاسخ سوال ۷

۷-۱_ بارگذاری داده‌ها و EDA

ابتدا داده‌ها را از فایل CSV آپلودشده در محیط Kaggle بارگذاری کرده و چند نمونه از داده‌ها را نمایش می‌دهیم:

```
df = pd.read_csv('/kaggle/input/customer-dateset/customers dataset.csv')
```

	Gender	Age	Income	Score
0	Male	19	15	39
1	Male	21	15	81
2	Female	20	16	6
3	Female	23	16	77
4	Female	31	17	40

سپس اطلاعات پایه‌ای مجموعه داده را بررسی می‌کنیم:

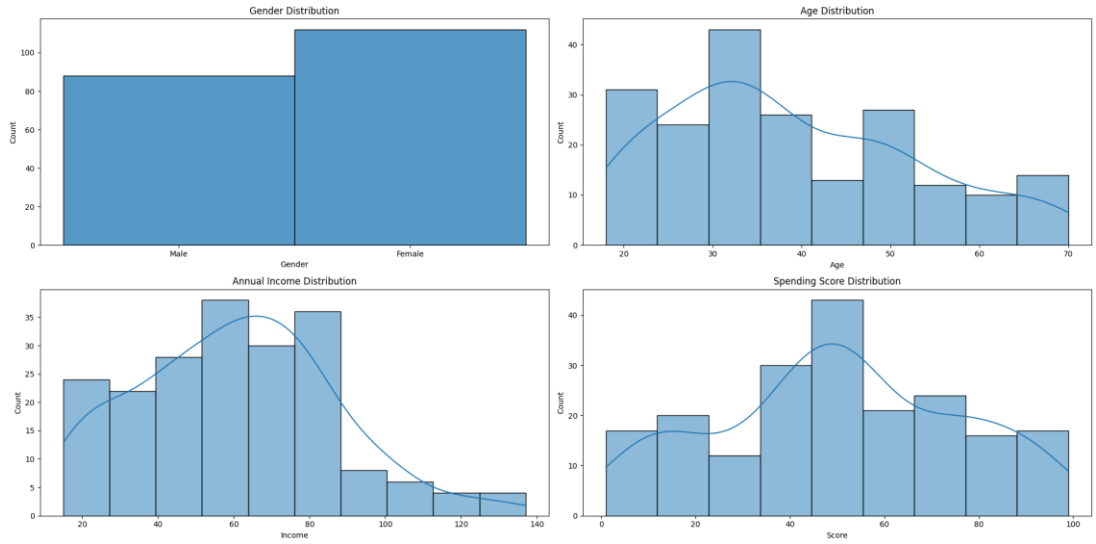
```
Basic information about the dataset:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 200 entries, 0 to 199
Data columns (total 4 columns):
 #   Column  Non-Null Count  Dtype 
---  ------  --------------  ----- 
 0   Gender  200 non-null    object
 1   Age     200 non-null    int64 
 2   Income  200 non-null    int64 
 3   Score   200 non-null    int64 
dtypes: int64(3), object(1)
memory usage: 6.4+ KB
None
```

این اطلاعات نشان می‌دهد که داده‌ها شامل ۲۰۰ رکورد و ۴ ستون (جنسیت، سن، درآمد و امتیاز) هستند و هیچ Missing Valueای در این داده‌ها وجود ندارد.

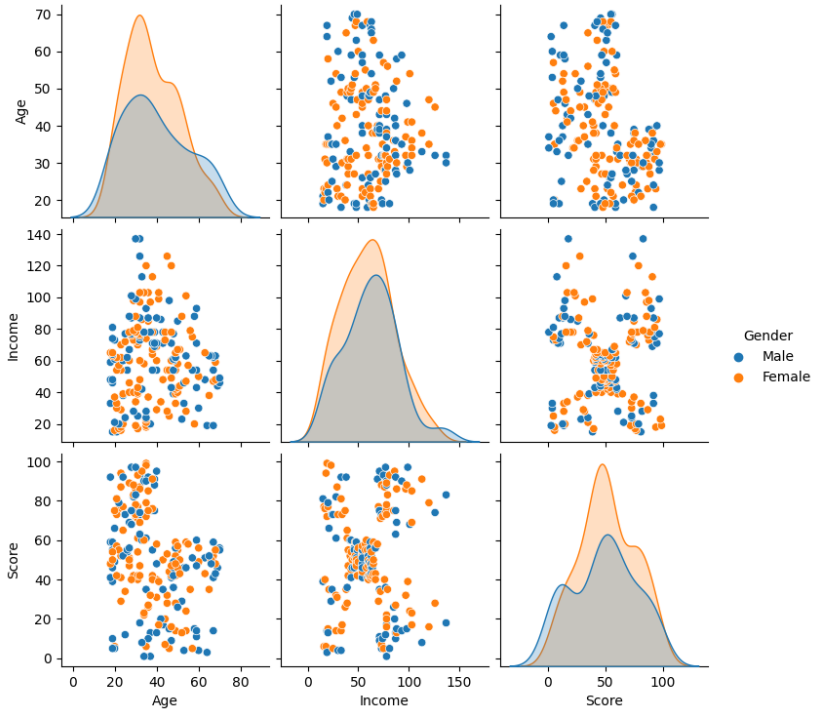
آماره‌های توصیفی مجموعه داده را نیز بررسی می‌کنیم:

```
Summary statistics of the dataset:
              Age      Income       Score
count  200.000000  200.000000  200.000000
mean    38.850000   60.560000   50.200000
std     13.969007   26.264721   25.823522
min     18.000000   15.000000    1.000000
25%     28.750000   41.500000   34.750000
50%     36.000000   61.500000   50.000000
75%     49.000000   78.000000   73.000000
max     70.000000  137.000000   99.000000
```

این آماره‌ها نشان می‌دهد که میانگین سن مشتریان ۳۸,۸۵ سال، میانگین درآمد ۶۰,۵۶ هزار دلار و میانگین امتیاز خرید ۵۰,۲ است.



شکل ۱۱ نمودارهای توزیع ویژگی‌ها



شکل ۱۲ نمودارهای توزیع Pairplot

بررسی توزیع ویژگی‌های مشتریان نشان می‌دهد که تفاوت‌های قابل توجهی بین ویژگی‌های مختلف و رابطه بین آن‌ها وجود دارد. این تفاوت‌ها می‌تواند در فرآیند خوشه‌بندی مشتریان مؤثر باشد. همچنین، نمودارهای pairplot به ما کمک می‌کنند تا الگوهای پیچیده‌تری را بین ویژگی‌های مختلف شناسایی کنیم. این اطلاعات می‌تواند به طراحی هر خوشه کمک کند.

۲-۷_ تبدیل ویژگی‌ها و استانداردسازی داده‌ها

تبدیل ویژگی 'Gender' به مقدار عددی

برای اینکه بتوانیم ویژگی Gender را به صورت عددی در الگوریتم‌های خوشه‌بندی استفاده کنیم، ابتدا این ویژگی را به مقادیر عددی تبدیل می‌کنیم. این کار با استفاده از LabelEncoder انجام می‌شود.

```
label_encoder = LabelEncoder()
df['Gender'] = label_encoder.fit_transform(df['Gender'])
```

در این کد، 'Gender' به صورت عددی تبدیل شده است، به طوری که 'Male' به ۱ و 'Female' به ۰ تبدیل شده است.

	Gender	Age	Income	Score
0	1	-1.424569	-1.738999	-0.434801
1	1	-1.281035	-1.738999	1.195704
2	0	-1.352802	-1.700830	-1.715913
3	0	-1.137502	-1.700830	1.040418
4	0	-0.563369	-1.662660	-0.395980

استانداردسازی داده‌ها

برای اطمینان از اینکه تمامی ویژگی‌ها در مقیاس یکسانی قرار دارند، ویژگی‌های "Age، Income" و 'Score' را استانداردسازی می‌کنیم. استانداردسازی باعث می‌شود میانگین هر ویژگی به ۰ و واریانس آن به ۱ تبدیل شود.

```
scaler = StandardScaler()
df[['Age', 'Income', 'Score']] = scaler.fit_transform(df[['Age', 'Income', 'Score']])
```

	Gender	Age	Income	Score
0	1	-1.424569	-1.738999	-0.434801
1	1	-1.281035	-1.738999	1.195704
2	0	-1.352802	-1.700830	-1.715913
3	0	-1.137502	-1.700830	1.040418
4	0	-0.563369	-1.662660	-0.395980

متغیرهای باینری مانند 'Gender' معمولاً نیاز به استانداردسازی ندارند. دلیل این امر این است که متغیرهای باینری تنها دو مقدار (مثلاً ۰ و ۱) دارند و مقیاس آنها از قبل ثابت است. استانداردسازی این متغیرها می‌تواند معنای اصلی آن‌ها را تغییر دهد و تفسیر نتایج را پیچیده کند.

۳-۷_ تعیین تعداد خوشه مناسب در خوشه‌بندی: روش‌های مختلف

در مسئله‌های خوشه‌بندی، یکی از چالش‌های اصلی تعیین تعداد مناسب خوشه‌هاست. انتخاب تعداد خوشه‌های نامناسب می‌تواند منجر به خوشه‌بندی ضعیف شود که اطلاعات ارزشمندی را از دست بدهد. در اینجا پنج روش برای تعیین تعداد خوشه مناسب را بررسی می‌کنیم:

K-means Distortion و تحلیل ELBOW

K-means Distortion:

در روش K-means، معیار Distortion یا همان مجموع مربعات خطا (SSE: Sum of Squared Errors) یکی از معیارهای مهم برای ارزیابی کیفیت خوشه‌بندی است. این معیار مقدار فاصله مربعی بین هر نقطه داده و مرکز خوشه مربوطه را محاسبه می‌کند و مجموع این فاصله‌ها را به عنوان Distortion یا SSE گزارش می‌دهد. هرچه تعداد خوشه‌ها بیشتر شود، مقدار Distortion کاهش می‌یابد، زیرا هر خوشه کوچک‌تر می‌شود و نقاط داده به مرکز خوشه نزدیک‌تر می‌شوند.

تحلیل ELBOW:

تحلیل Elbow یک روش گرافیکی برای تعیین تعداد خوشه بهینه است. در این روش، تعداد خوشه‌های مختلف را امتحان می‌کنیم و مقدار SSE مربوط به هر تعداد خوشه را محاسبه می‌کنیم.سپس یک نمودار رسم می‌کنیم که تعداد خوشه‌ها را در محور افقی و مقدار SSE را در محور عمودی نشان می‌دهد.

نقطه‌ای که کاهش مقدار SSE به طور قابل توجهی کاهش می‌یابد و نمودار به شکل یک آرنج (elbow) درمی‌آید، به عنوان تعداد خوشه بهینه انتخاب می‌شود. این نقطه نشان می‌دهد که افزودن خوشه‌های بیشتر بهبود قابل توجهی در کیفیت خوشه‌بندی ایجاد نمی‌کند.

مزایا: روش Elbow ساده و بصری است و به راحتی می‌توان تعداد مناسب خوشه‌ها را از روی نمودار تشخیص داد.

معایب: در برخی موارد، نمودار Elbow ممکن است شکل واضحی نداشته باشد یا چندین نقطه Elbow وجود داشته باشد که تصمیم‌گیری را مشکل می‌کند.

این روش بر اساس معیار SSE عمل می‌کند که ممکن است به تنهایی نتواند تمامی جنبه‌های کیفیت خوشه‌بندی را در نظر بگیرد.

Silhouette Score

Silhouette Score یک معیار اندازه‌گیری کیفیت خوشه‌بندی است که میزان نزدیکی هر نقطه به خوشه خود و دوری آن از خوشه‌های دیگر را محاسبه می‌کند. این معیار برای هر نقطه داده محاسبه می‌شود و مقدار میانگین آن برای تمامی نقاط داده به عنوان Silhouette Score نهایی در نظر گرفته می‌شود.

مقدار Silhouette بین -۱ و ۱ قرار دارد.

مقدار نزدیک به ۱ نشان می‌دهد که نقاط داده به خوشه خود نزدیک و از خوشه‌های دیگر دور هستند.

مقدار نزدیک به ۰ نشان می‌دهد که نقاط داده در مرز بین خوشه‌ها قرار دارند.

مقدار منفی نشان می‌دهد که نقاط داده به خوشه‌های دیگر نزدیک‌تر از خوشه خود هستند که نشان‌دهنده خوشه‌بندی نامناسب است.

تعداد خوشه‌هایی که بیشترین Silhouette Score را دارند به عنوان تعداد خوشه بهینه انتخاب می‌شوند.

مزایا: Silhouette Score به طور مستقیم کیفیت خوشه‌بندی را بر اساس نزدیکی و دوری نقاط به خوشه‌ها ارزیابی می‌کند و معیار جامعی برای ارزیابی است.

معایب: محاسبه Silhouette Score برای داده‌های بزرگ ممکن است محاسباتی سنگین باشد و به منابع بیشتری نیاز داشته باشد.

Davies-Bouldin Index

شاخص Davies-Bouldin (DBI) برای ارزیابی کیفیت خوشه‌بندی بر اساس نسبت فاصله بین خوشه‌ها به قطر خوشه‌ها استفاده می‌شود. این شاخص میانگین نزدیک‌ترین فاصله بین خوشه‌ها را در نظر می‌گیرد و با تقسیم فاصله بین مراکز خوشه‌ها بر قطر خوشه‌ها محاسبه می‌شود.

هرچه مقدار Davies-Bouldin Index کمتر باشد، خوشه‌ها بهتر از هم تفکیک شده‌اند.

فرمول محاسبه DBI به صورت زیر است:

$$\mathrm{DB}=\frac{1}{N}\sum_{i=1}^{N} D_i$$
$$D_i=\max_{j:i\neq j} R_{i,j}$$
$$R_{i,j}=\frac{S_i+S_j}{M_{i,j}}$$

or

$$DB=\frac{1}{N}\sum_{i=1}^{N}\max_{i\neq j}\left(\frac{S_i+S_j}{d(c_i,c_j)}\right)$$

k تعداد خوشه‌هاست. Si و Sj میانگین فاصله نقاط داده در خوشه i به مرکز خوشه است. d ij فاصله بین مراکز خوشه‌های i و j است. مقادیر کوچکتر DBI نشان‌دهنده خوشه‌بندی بهتر است.

مزایا: Davies-Bouldin Index از هر دو معیار فشردگی داخل خوشه‌ها و جداپذیری بین خوشه‌ها استفاده می‌کند که معیار جامعی برای ارزیابی خوشه‌بندی است.

معایب: این شاخص به تعداد خوشه‌ها حساس است و ممکن است با افزایش تعداد خوشه‌ها مقدار آن بهبود یابد، حتی اگر کیفیت خوشه‌بندی کلی کاهش یابد.

Calinski-Harabasz Index:

شاخص Calinski-Harabasz (CH) یا نسبت واریانس بین خوشه‌ها به واریانس داخل خوشه‌ها یکی دیگر از معیارهای ارزیابی کیفیت خوشه‌بندی است. این شاخص به عنوان نسبت مجموع مربعات بین خوشه‌ها به مجموع مربعات داخل خوشه‌ها ضرب در نسبت تعداد نقاط داده‌ها به تعداد خوشه‌ها محاسبه می‌شود.

هرچه مقدار Calinski-Harabasz Index بیشتر باشد، کیفیت خوشه‌بندی بهتر است.

فرمول محاسبه CH به صورت زیر است:

$$s = \frac{\mathrm{tr}(B_k)}{\mathrm{tr}(W_k)} \times \frac{n_E - k}{k - 1}$$

tr(Bk) مجموع مربعات بین خوشه‌هاست.(tr(Wk مجموع مربعات داخل خوشه‌هاست. k تعداد خوشه‌هاست. n تعداد نقاط داده است.

مزایا: Calinski-Harabasz Index یک شاخص قوی برای ارزیابی کیفیت خوشه‌بندی است و به طور گسترده در کاربردهای مختلف استفاده می‌شود.

معایب: این شاخص به تعداد خوشه‌ها حساس است و ممکن است با افزایش تعداد خوشه‌ها مقدار آن بهبود یابد، حتی اگر کیفیت خوشه‌بندی کلی کاهش یابد.

Dunn Index

شاخص Dunn (DI) برای شناسایی خوشه‌های فشرده و جدا استفاده می‌شود. این شاخص نسبت کوچک‌ترین فاصله بین نقاط داده در خوشه‌های مختلف به بزرگ‌ترین قطر خوشه‌ها را اندازه‌گیری می‌کند.

فرمول محاسبه DI به صورت زیر است:

$$D = \frac{min_{1 \le i < j \le n} d(i,j)}{max_{1 \le k \le n} d'(k)}$$

d(i,j) کوچک‌ترین فاصله بین نقاط داده در خوشه‌های مختلف i و j است. d'(k) بزرگ‌ترین قطر خوشه l است که بیشترین فاصله بین نقاط داده در همان خوشه را اندازه‌گیری می‌کند.

مقدار بیشتر شاخص Dunn نشان‌دهنده خوشه‌بندی بهتر است.

مزایا: Dunn Index به طور مستقیم بر جداسازی بین خوشه‌ها و فشردگی داخل خوشه‌ها تمرکز دارد و می‌تواند به خوبی خوشه‌های مناسب را شناسایی کند.

معایب: محاسبه Dunn Index برای داده‌های بزرگ ممکن است محاسباتی سنگین باشد و به منابع بیشتری نیاز داشته باشد.

۴-۷_ نتایج روش‌های مختلف برای تعیین تعداد خوشه مناسب

اجرای الگوریتم K-means برای تعداد مختلف خوشه‌ها

در این بخش، الگوریتم K-means را برای تعداد خوشه‌های مختلف (از ۲ تا ۱۰) اجرا کرده و معیارهای مختلف ارزیابی را محاسبه می‌کنیم:

```
cluster_range = range(2, 11)
inertia = []
silhouette_scores = []
davies_bouldin_indices = []
calinski_harabasz_indices = []

for k in cluster_range:
    kmeans = KMeans(n_clusters=k, random_state=42)
    kmeans.fit(df)
    labels = kmeans.labels_

    inertia.append(kmeans.inertia_)
    silhouette_scores.append(silhouette_score(df, labels))
    davies_bouldin_indices.append(davies_bouldin_score(df, labels))
    calinski_harabasz_indices.append(calinski_harabasz_score(df, labels))
```

شاخص Dunn در کتابخانه یافت نشد پس آن را به صورت دستی محاسبه می‌کنیم. این شاخص نسبت کوچک‌ترین فاصله بین خوشه‌ها به بزرگ‌ترین فاصله داخل خوشه‌ها را اندازه‌گیری می‌کند:

```
def dunn_index(data, labels):
  unique_clusters = np.unique(labels)
  max_intra_cluster_dist = 0
  min_inter_cluster_dist = float('inf')

  for cluster in unique_clusters:
    cluster_points = data[labels == cluster]

    for i in range(len(cluster_points)):
      for j in range(i + 1, len(cluster_points)):
        dist = euclidean(cluster_points[i], cluster_points[j])
        if dist > max_intra_cluster_dist:
          max_intra_cluster_dist = dist
  for i in range(len(unique_clusters)):
    for j in range(i + 1, len(unique_clusters)):
      cluster_i = data[labels == unique_clusters[i]]
      cluster_j = data[labels == unique_clusters[j]]

      for point_i in cluster_i:
        for point_j in cluster_j:
          dist = euclidean(point_i, point_j)
          if dist < min_inter_cluster_dist:
            min_inter_cluster_dist = dist

  return min_inter_cluster_dist / max_intra_cluster_dist
```

محاسبه بزرگ‌ترین فاصله داخل خوشه‌ها:

این مقدار نشان‌دهنده فشردگی هر خوشه است. هرچه این مقدار کمتر باشد، نقاط داده در داخل خوشه به هم نزدیک‌تر هستند و خوشه فشرده‌تر است.با استفاده از دو حلقه تو در تو، فاصله اقلیدسی بین هر جفت نقطه در هر خوشه محاسبه می‌شود.

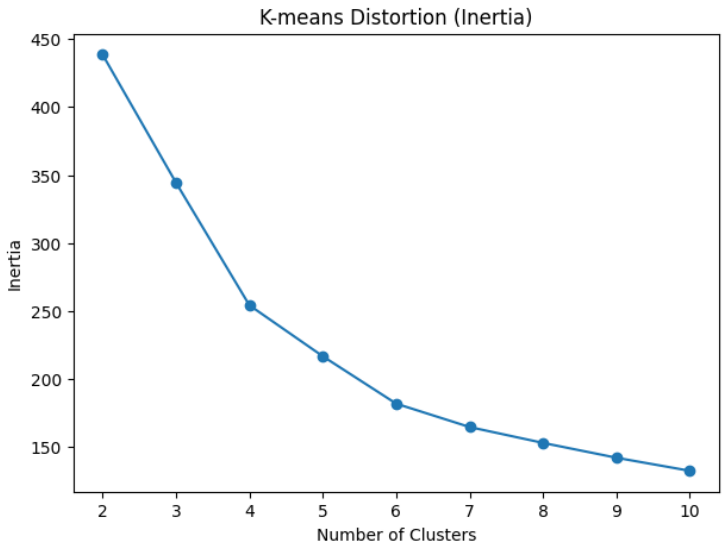
اگر این فاصله بیشتر از max_intra_cluster_dist باشد، مقدار max_intra_cluster_dist به‌روزرسانی می‌شود.

محاسبه کوچک‌ترین فاصله بین خوشه‌ها:

ین مقدار نشان‌دهنده جدایی بین خوشه‌هاست. هرچه این مقدار بیشتر باشد، خوشه‌ها از هم جداتر هستند و خوشه‌بندی بهتر است. با استفاده از دو حلقه تو در تو، فاصله اقلیدسی بین هر جفت خوشه محاسبه می‌شود.

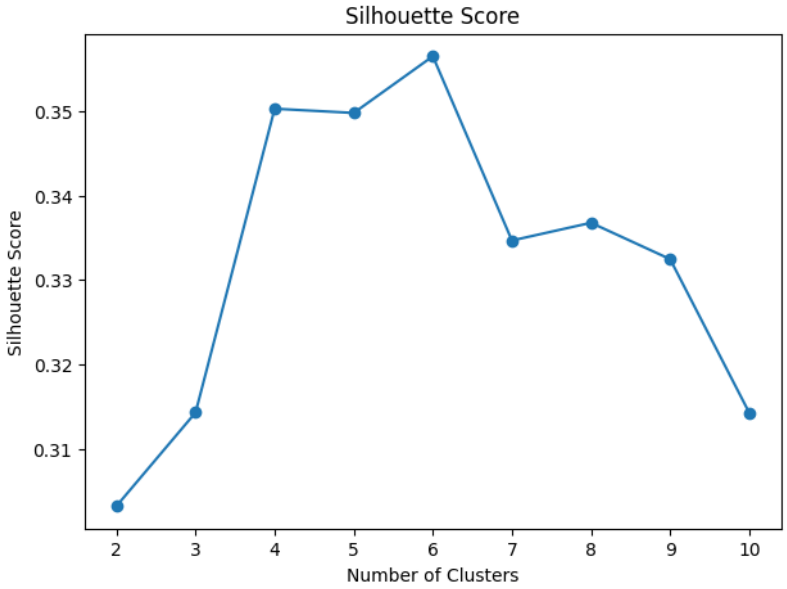
اگر این فاصله کمتر از min_inter_cluster_dist باشد، مقدار min_inter_cluster_dist به‌روزرسانی می‌شود.

نمودار K-means Distortion و تحلیل ELBOW



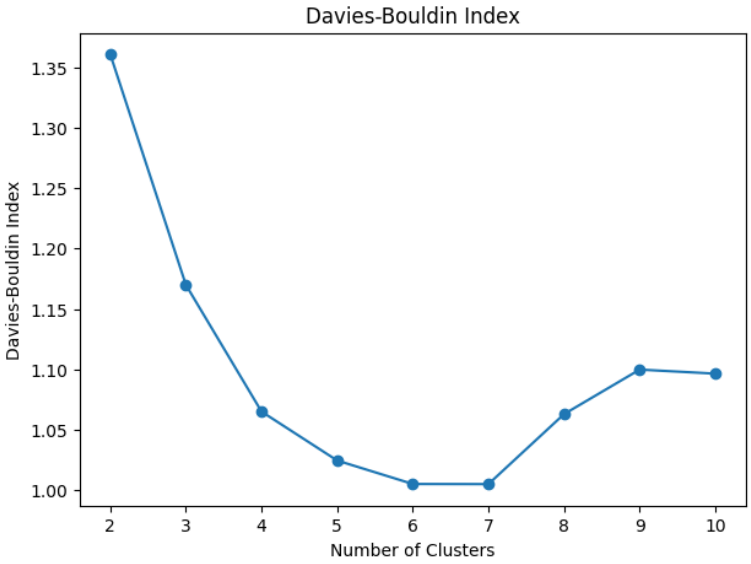
در نمودار Elbow، تعداد خوشه‌ها در محور افقی و مقدار Inertia (SSE) در محور عمودی نشان داده شده است. کاهش قابل توجهی در مقدار Inertia از ۲ تا ۴ خوشه مشاهده می‌شود. از ۵ خوشه به بعد، کاهش Inertia به تدریج کمتر می‌شود و نمودار به شکل یک آرنج درمی‌آید. این نقطه آرنج (Elbow) نشان می‌دهد که تعداد خوشه‌های مناسب بین ۴ و ۵ است.

نمودار Silhouette Score



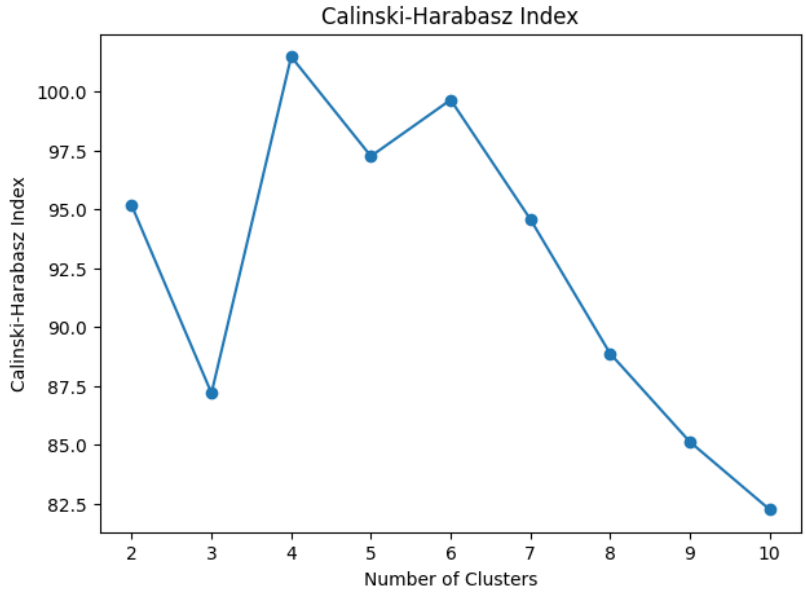
در نمودار Silhouette Score، تعداد خوشه‌ها در محور افقی و مقدار Silhouette Score در محور عمودی نشان داده شده است. بیشترین مقدار Silhouette Score برای ۶ خوشه مشاهده می‌شود. این نشان می‌دهد که خوشه‌بندی با ۶ خوشه بهترین کیفیت را دارد.

نمودار Davies-Bouldin Index



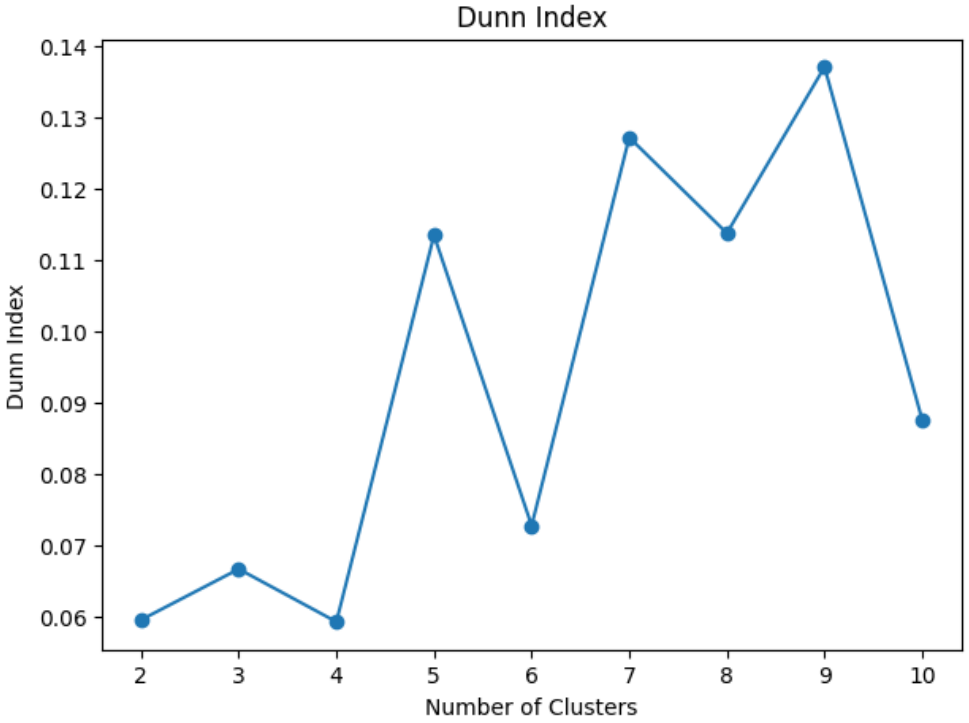
در نمودار Davies-Bouldin Index، تعداد خوشه‌ها در محور افقی و مقدار DBI در محور عمودی نشان داده شده است. کمترین مقدار DBI برای ۶ خوشه مشاهده می‌شود. هرچه مقدار DBI کمتر باشد، کیفیت خوشه‌بندی بهتر است.

نمودار Calinski-Harabasz Index



در نمودار Calinski-Harabasz Index، تعداد خوشه‌ها در محور افقی و مقدار CH در محور عمودی نشان داده شده است. بیشترین مقدار CH برای ۴ خوشه مشاهده می‌شود. هرچه مقدار CH بیشتر باشد، کیفیت خوشه‌بندی بهتر است.

نمودار Dunn Index



در نمودار Dunn Index، تعداد خوشه‌ها در محور افقی و مقدار DI در محور عمودی نشان داده شده است. بیشترین مقدار DI برای ۹ خوشه مشاهده می‌شود. هرچه مقدار DI بیشتر باشد، کیفیت خوشه‌بندی بهتر است.

برای اجرای خوشه‌بندی نهایی، تصمیم گرفتم تعداد خوشه‌های ۴، ۶ و ۹ را بررسی کنم و نتایج خوشه‌بندی را مقایسه نمایم. با توجه به کیفیت خوشه‌بندی و نیازهای خاص مسئله، بهترین تعداد خوشه را انتخاب خواهم کرد. استفاده از این سه تعداد خوشه به من امکان می‌دهد تا به تحلیل جامع‌تری دست پیدا کنیم .

اگر باید تنها یک تعداد خوشه انتخاب کنیم، تعداد ۶ خوشه را به دلیل اشتراک بیشتر معیارهای کیفیت خوشه‌بندی ترجیح می‌دهم.

اما بررسی نتایج برای ۴ و ۹ خوشه نیز می‌تواند دیدگاه‌های مفیدی ارائه دهد و به انتخاب بهینه خوشه‌بندی کمک کند.

۵-۷_ نمایش داده‌های با ابعاد بالا: روش‌های مختلف

داده‌های با ابعاد بالا به دلیل پیچیدگی و تعداد بالای ویژگی‌ها به راحتی قابل نمایش و تفسیر نیستند. برای نمایش و تحلیل این داده‌ها از روش‌های کاهش ابعاد استفاده می‌شود. در ادامه به بررسی چندین روش متداول برای کاهش ابعاد و نمایش داده‌ها می‌پردازیم.

تحلیل مولفه‌های اصلی (PCA)

تحلیل مولفه‌های اصلی (PCA) یکی از پرکاربردترین روش‌های کاهش ابعاد است که با تبدیل داده‌ها به یک فضای جدید با استفاده از ترکیبی خطی از ویژگی‌های اصلی، ابعاد داده‌ها را کاهش می‌دهد. این روش بر اساس حفظ بیشترین واریانس داده‌ها در فضای جدید عمل می‌کند.

مراحل انجام PCA:

- **استانداردسازی داده‌ها:** ابتدا داده‌ها نرمال استاندارد می‌شوند که هر ویژگی دارای میانگین صفر و واریانس یک باشد. این کار کمک می‌کند تا همه ویژگی‌ها در یک مقیاس باشند.
- **محاسبه ماتریس کوواریانس:** این ماتریس نشان می‌دهد که چطور ویژگی‌های مختلف به یکدیگر مرتبط هستند.
- **محاسبه بردارهای ویژه و مقادیر ویژه:** این بردارها و مقادیر به ما کمک می‌کنند تا جهت‌هایی را که داده‌ها در آن‌ها بیشترین تغییرات را دارند پیدا کنیم.
- **انتخاب مولفه‌های اصلی:** از بین بردارهای ویژه، آن‌هایی که بیشترین مقادیر ویژه را دارند انتخاب می‌شوند. این‌ها مولفه‌های اصلی ما هستند.
- **تبدیل داده‌ها:** داده‌های اصلی به این مولفه‌های اصلی تبدیل می‌شوند. به این ترتیب، تعداد ویژگی‌های داده‌ها کاهش می‌یابد.

مزایا:

حفظ بیشترین واریانس داده‌ها.

کاهش ابعاد به صورت خطی و قابل تفسیر.

معایب:

ناتوانی در تشخیص روابط غیرخطی بین ویژگی‌ها.

در صورت وجود نویز در داده‌ها، ممکن است نتایج تحت تأثیر قرار گیرد.

الگوریتم t-SNE

t-SNE (t-Distributed Stochastic Neighbor Embedding) یک روش غیرخطی برای کاهش ابعاد و نمایش داده‌های با ابعاد بالا است. این روش با حفظ ساختار محلی داده‌ها در فضای جدید، به نمایش بهتر داده‌ها در ابعاد پایین می‌پردازد. . این روش با حفظ ساختار محلی داده‌ها، به نمایش بصری بهتر داده‌ها کمک می‌کند. t-SNE داده‌ها را به یک فضای با ابعاد کمتر (معمولاً دو یا سه بعد) تبدیل می‌کند که در آن داده‌هایی که در فضای اصلی به هم نزدیک هستند، در فضای جدید نیز به هم نزدیک باقی می‌مانند.

مراحل اصلی t-SNE:

محاسبه احتمالات شباهت در فضای اصلی: t-SNE با استفاده از یک تابع Gauss، احتمالات شباهت بین نقاط داده را محاسبه می‌کند.

محاسبه احتمالات شباهت در فضای جدید: در فضای با ابعاد کمتر، t-SNE با استفاده از توزیع t، احتمالات شباهت بین نقاط را محاسبه می‌کند.

بهینه‌سازی: t-SNE سعی می‌کند با حداقل کردن اختلاف بین این دو مجموعه احتمال، داده‌ها را در فضای جدید قرار دهد.

مزایا:

حفظ ساختار محلی داده‌ها.

نمایش بصری مناسب داده‌های پیچیده.

معایب:

نیاز به تنظیم پارامترهای مختلف.

زمان محاسباتی بالا برای داده‌های بزرگ.

تحلیل تفکیکی خطی (LDA)

تحلیل تفکیکی خطی (LDA) روشی است که به کاهش ابعاد و تفکیک بهتر کلاس‌ها می‌پردازد. این روش بر اساس حداکثرسازی نسبت بین واریانس بین کلاسی به واریانس داخل کلاسی عمل می‌کند.

مراحل انجام LDA:

محاسبه میانگین هر کلاس: میانگین هر ویژگی برای هر کلاس محاسبه می‌شود.

محاسبه ماتریس‌های پراکندگی بین کلاسی و داخل کلاسی: ماتریس پراکندگی بین کلاسی و داخل کلاسی محاسبه می‌شود.

محاسبه مقادیر و بردارهای ویژه: مقادیر و بردارهای ویژه ماتریس پراکندگی محاسبه می‌شوند.

انتخاب مولفه‌های تفکیکی: بردارهای ویژه انتخاب شده و داده‌ها به فضای جدید انتقال می‌یابند.

مزایا:

افزایش قدرت تفکیک بین کلاس‌ها.

کاهش ابعاد با حفظ اطلاعات کلاس‌ها.

معایب:

نیاز به داشتن برچسب‌های کلاس برای داده‌ها.

ناتوانی در تشخیص روابط غیرخطی بین ویژگی‌ها.

تحلیل اجزای مستقل (ICA)

تجزیه مؤلفه‌های مستقل (ICA) یک تکنیک پردازش سیگنال است که برای جدا کردن منابع مستقل از یک سیگنال ترکیبی استفاده می‌شود. هدف ICA شناسایی و استخراج سیگنال‌های مستقل از داده‌های چندمتغیره است. این روش به ویژه برای تحلیل داده‌های EEG که شامل ترکیبی از سیگنال‌های مختلف مغزی هستند، بسیار مفید است.

مراحل انجام ICA:

مرکزسازی داده‌ها: ابتدا میانگین داده‌ها صفر می‌شود. این کار کمک می‌کند تا ICA بهتر عمل کند.

سفیدسازی داده‌ها: داده‌ها به گونه‌ای تغییر می‌کنند که همبستگی بین ویژگی‌ها حذف شود و واریانس هر ویژگی یکسان شود.

یافتن سیگنال‌های مستقل: با استفاده از الگوریتم ICA، سیگنال‌های مستقل از داده‌ها استخراج می‌شوند.

مزایا:

تشخیص و جداسازی منابع مستقل در داده‌ها.

کاربردهای متعدد در پردازش سیگنال و تصاویر.

معایب:

نیاز به تعیین تعداد اجزای مستقل.

حساسیت به نویز و ناپایداری در برخی موارد.

UMAP

UMAP (Uniform Manifold Approximation and Projection) یک روش کاهش ابعاد غیرخطی است که برای نمایش داده‌های با ابعاد بالا به کار می‌رود. این روش بر اساس نظریه منیفولد و تئوری توپولوژی عمل می‌کند و تلاش می‌کند تا ساختارهای محلی و جهانی داده‌ها را حفظ کند.

مراحل انجام UMAP:

ساخت گراف نزدیکی: یک گراف نزدیکی برای داده‌ها ساخته می‌شود که روابط محلی را نشان می‌دهد.

تخمین ساختار منیفولد: ساختار منیفولد با استفاده از گراف نزدیکی تخمین زده می‌شود.

نگاشت به فضای کم‌بعد: داده‌ها به فضای کم‌بعد نگاشت می‌شوند و تلاش می‌شود تا ساختارهای محلی و جهانی حفظ شوند.

مزایا:

حفظ ساختارهای محلی و جهانی داده‌ها.

کارایی بالا و سرعت بیشتر نسبت به t-SNE.

قابلیت کار با داده‌های بزرگ و پیچیده.

معایب:

نیاز به تنظیم پارامترهای مختلف.

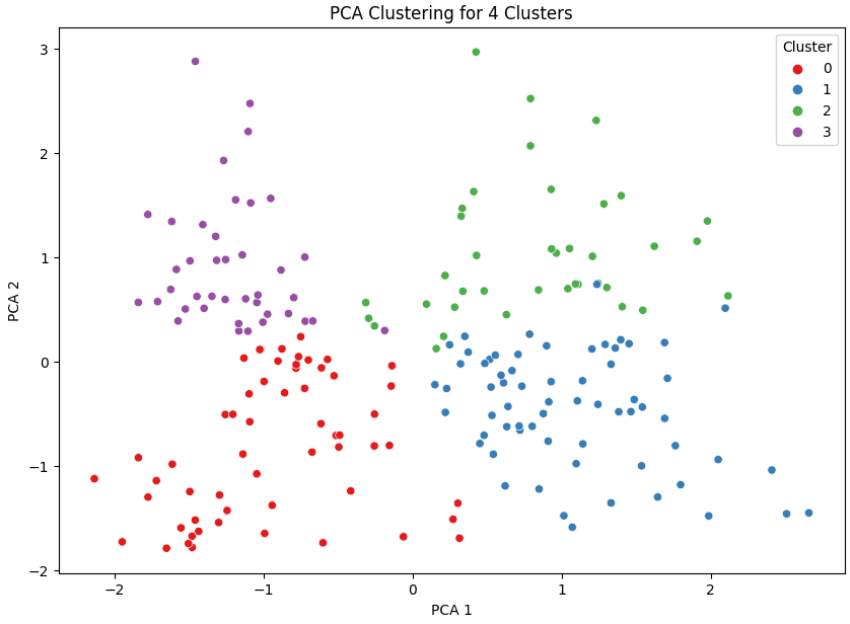
در برخی موارد، تفسیر نتایج ممکن است پیچیده باشد.

با توجه به ویژگی‌های داده‌ها و نیازهای خاص مسئله، در اینجا دو روش **PCA** و **t-SNE** را انتخاب می‌کنیم. این دو روش توانایی خوبی در کاهش ابعاد و نمایش داده‌ها دارند و تحلیل تأثیر هر ویژگی در خوشه‌بندی را ممکن می‌سازند.

۶-۷_ تحلیل نتایج خوشه‌بندی با استفاده از PCA

در این تحلیل، از روش PCA برای کاهش ابعاد داده‌ها استفاده کردیم و سپس داده‌ها را با استفاده از الگوریتم K-means برای تعداد خوشه‌های مختلف (۴، ۶ و ۹) خوشه‌بندی کردیم. در ادامه، نتایج هر خوشه‌بندی را با نمودارهای حاصل از PCA مورد بررسی قرار می‌دهیم.

خوشه‌بندی با ۴ خوشه



شکل ۱۳ خوشه‌بندی با ۴ خوشه

در این نمودار، داده‌ها در فضای دو بعدی با استفاده از PCA نمایش داده شده‌اند و هر خوشه با رنگی متفاوت مشخص شده است.

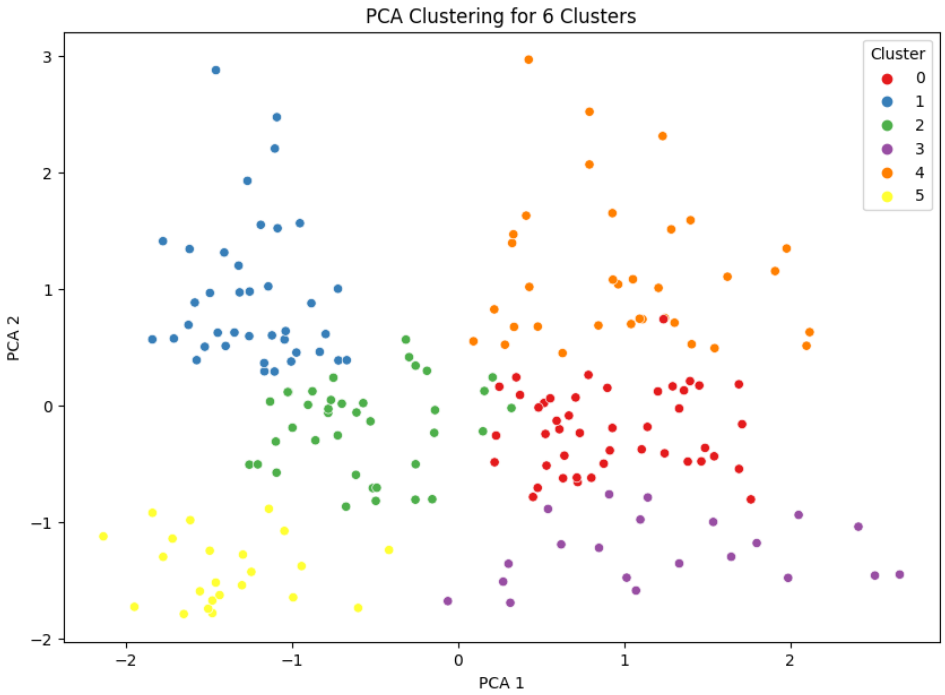
خوشه ۰: این خوشه شامل نقاطی با ویژگی‌های خاص است که به طور مشخص از دیگر خوشه‌ها متمایز است.

خوشه ۱: این خوشه نیز به وضوح از دیگر خوشه‌ها جدا شده است.

خوشه ۲: این خوشه دارای نقاطی است که نسبتاً متراکم هستند و از خوشه‌های دیگر جدا شده‌اند.

خوشه ۳: این خوشه نیز با تراکم خوبی از نقاط تشکیل شده است و از دیگر خوشه‌ها متمایز است.

خوشه‌بندی با ۶ خوشه



شکل ۱۴ خوشه‌بندی با ۶ خوشه

خوشه ۰: این خوشه شامل نقاطی با ویژگی‌های خاص است که به طور مشخص از دیگر خوشه‌ها متمایز است.

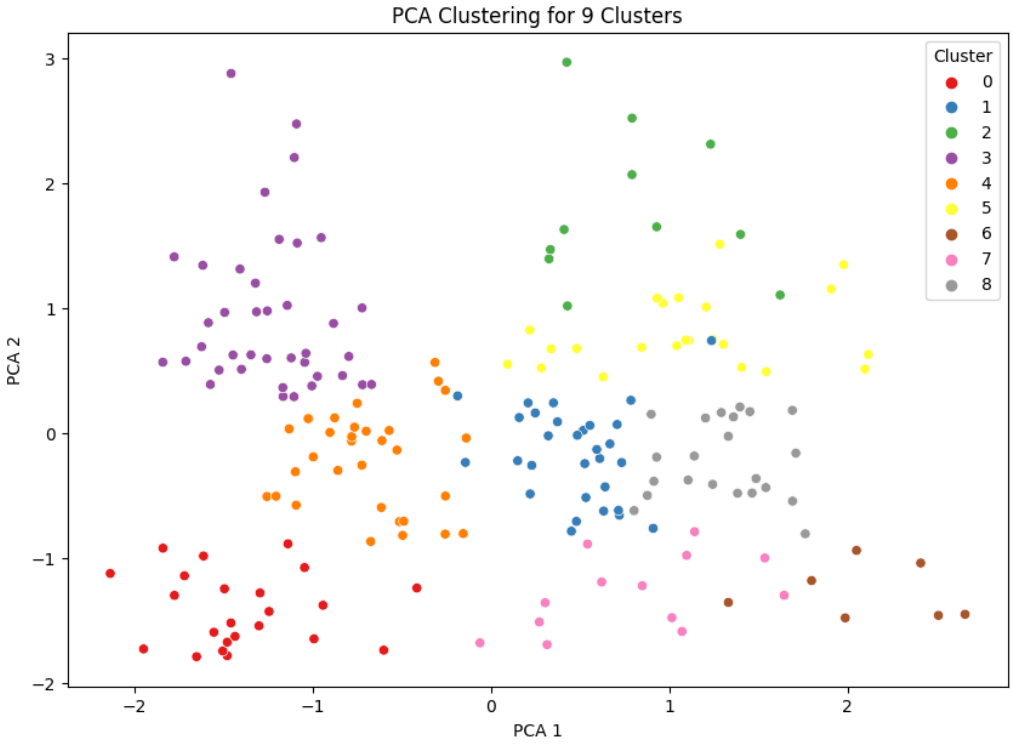
خوشه ۱: این خوشه نیز به وضوح از دیگر خوشه‌ها جدا شده است.

خوشه ۲: این خوشه دارای نقاطی است که نسبتاً متراکم هستند و از خوشه‌های دیگر جدا شده‌اند.

خوشه ۳: این خوشه نیز با تراکم خوبی از نقاط تشکیل شده است و از دیگر خوشه‌ها متمایز است.

خوشه ۴ و ۵: این خوشه‌ها نیز به خوبی از سایر خوشه‌ها جدا شده‌اند و هر یک دارای ویژگی‌های منحصر به فرد خود هستند.

نتایج خوشه‌بندی با ۹ خوشه



شکل ۱۵ نتایج خوشه‌بندی با ۹ خوشه

خوشه‌های ۰ تا ۸: هر یک از این خوشه‌ها به وضوح از دیگر خوشه‌ها جدا شده‌اند و هر خوشه دارای ویژگی‌های منحصر به فرد خود است. تراکم نقاط در هر خوشه متفاوت است و این نشان‌دهنده تنوع بیشتر در داده‌ها است.

۴ خوشه: برای تفکیک کلی و ساده‌تر داده‌ها مناسب است.

۶ خوشه: برای تحلیل دقیق‌تر و شناسایی ویژگی‌های بیشتر در داده‌ها مناسب است.

۹ خوشه: برای تحلیل جامع‌تر و شناسایی جزئیات بیشتر در داده‌ها مناسب است.

با توجه به نیازهای خاص پروژه، می‌توان از هر یک از این تعداد خوشه‌ها استفاده کرد. اگر نیاز به تحلیل دقیق‌تر و جزئی‌تر داده‌ها داریم، تعداد خوشه‌های بیشتر (۶ یا ۹) توصیه می‌شود.

تأثیر ویژگی‌ها در خوشه‌بندی با استفاده از PCA

در این بخش، داده‌های مشتریان را با استفاده از PCA به دو مولفه اصلی کاهش داده‌ایم. پس از اجرای PCA، اثرگذاری هر ویژگی در مولفه‌های اصلی را استخراج کرده و تحلیل می‌کنیم که هر ویژگی چقدر در این مولفه‌های اصلی تأثیر دارد.

استخراج اثرگذاری ویژگی‌ها – PCA Loadings

اثرگذاری هر ویژگی نشان‌دهنده تأثیر آن ویژگی در مولفه‌های اصلی هستند. به عبارتی، این مقادیر نشان می‌دهند که هر ویژگی به چه میزان در شکل‌گیری هر مولفه اصلی سهیم است.

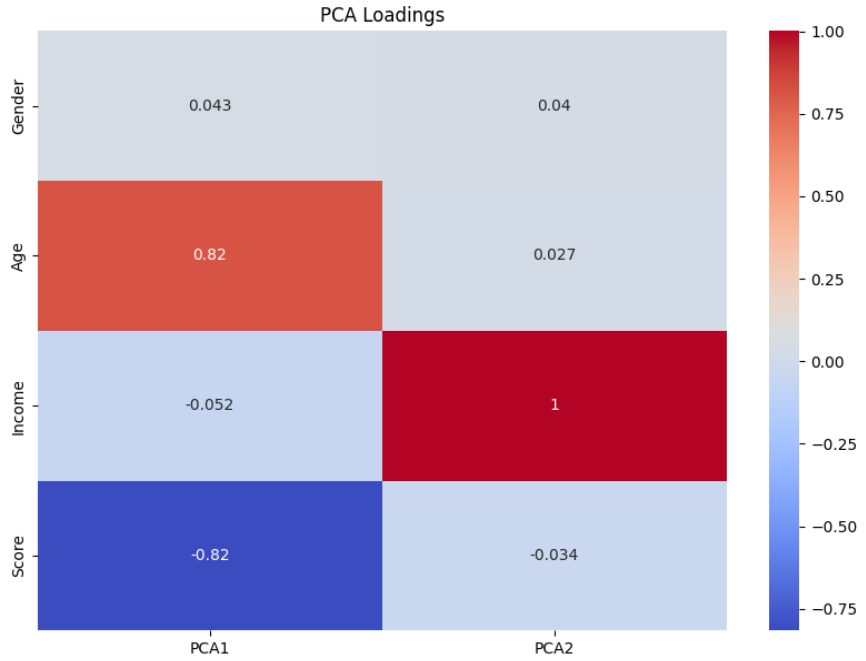
```
loadings = pca.components_.T * np.sqrt(pca.explained_variance_)

loadings_df = pd.DataFrame(loadings, columns=['PCA1', 'PCA2'], index=['Gender', 'Age',
'Income', 'Score'])
```

مقادیر اثرگذاری به صورت زیر است:

```
PCA Loadings:
            PCA1      PCA2
Gender  0.043161  0.039560
Age     0.816125  0.026852
Income -0.051561  1.000999
Score  -0.815839 -0.034309
```

برای نمایش بهتر و تفسیر آسان‌تر اثرگذاری ویژگی‌ها، از یک نمودار heatmap استفاده کرده‌ایم. این نمودار به وضوح نشان می‌دهد که کدام ویژگی‌ها در هر مولفه اصلی بیشترین تأثیر را دارند.



شکل ۱۶ اثرگذاری ویژگی‌ها – PCA Loadings

PCA1:

Age: دارای اثر مثبت بزرگ (۰,۸۲) است که نشان می‌دهد این مولفه اصلی به شدت تحت تأثیر سن قرار دارد.

Score: دارای اثر منفی بزرگ (-۰,۸۲) است که نشان می‌دهد این مولفه اصلی به شدت تحت تأثیر امتیاز خرید قرار دارد.

Gender و **Income**: دارای اثرات بسیار کوچکی هستند که نشان می‌دهد این ویژگی‌ها تأثیر کمی بر مولفه اول دارند.

این مولفه عمدتاً تحت تأثیر Age و Score قرار دارد. به عبارت دیگر، تفاوت‌های سنی و امتیازات خرید بیشترین تأثیر را در جداسازی خوشه‌ها در این بعد دارند.

PCA2:

Income: دارای اثر بسیار بزرگ (۱,۰۰) است که نشان می‌دهد این مولفه اصلی به شدت تحت تأثیر درآمد قرار دارد.

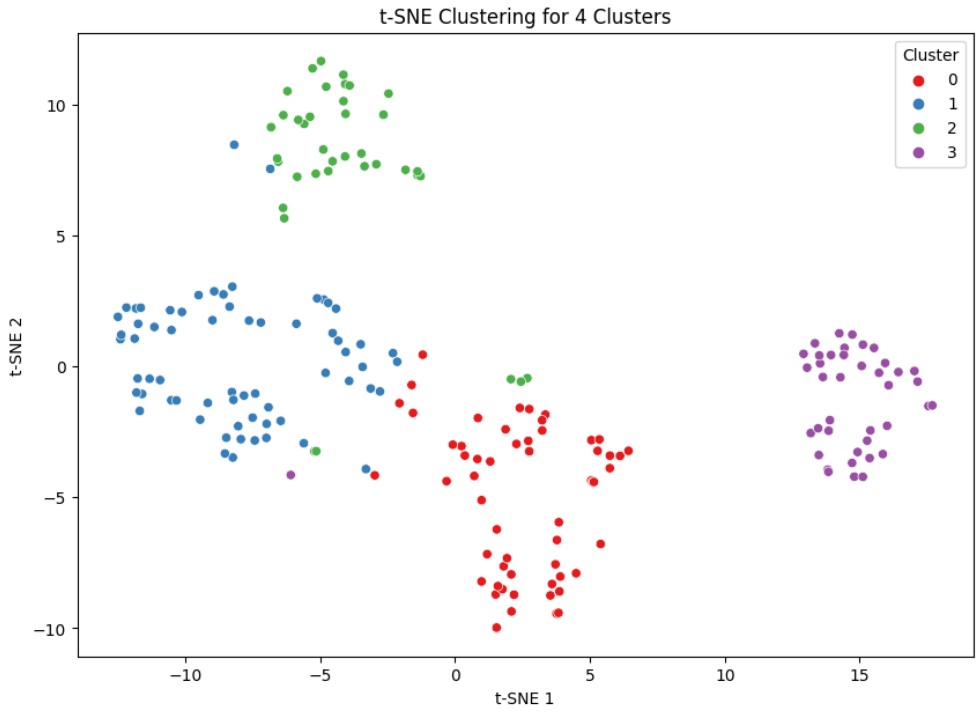
Gender، **Age** و **Score**: دارای اثرات بسیار کوچکی هستند که نشان می‌دهد این ویژگی‌ها تأثیر کمی بر مولفه دوم دارند.

این مولفه عمدتاً تحت تأثیر Income قرار دارد. به عبارت دیگر، تفاوت‌های درآمدی بیشترین تأثیر را در جداسازی خوشه‌ها در این بعد دارند.

۷-۷_ تحلیل نتایج خوشه‌بندی با استفاده از t-SNE

در این قسمت از تمرین، از t-SNE برای کاهش ابعاد داده‌های مشتریان استفاده کردیم و سپس داده‌ها را با استفاده از الگوریتم K-means برای تعداد خوشه‌های مختلف (۴، ۶ و ۹) خوشه‌بندی کردیم. نتایج هر خوشه‌بندی را با نمودارهای حاصل از t-SNE مورد بررسی قرار می‌دهیم.

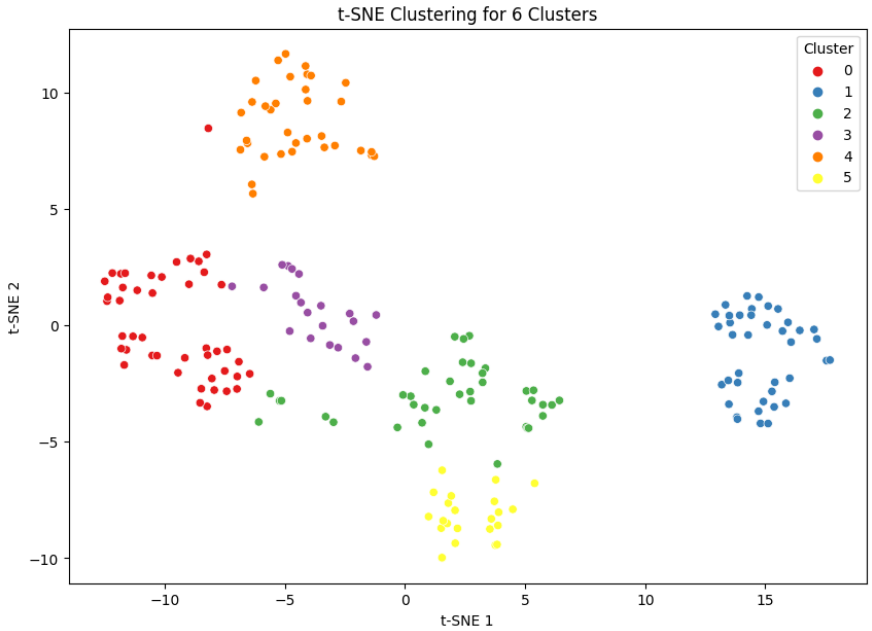
خوشه‌بندی با ۴ خوشه



شکل ۱۷ نتایج خوشه‌بندی با ۴ خوشه

خوشه‌ها به خوبی از یکدیگر جدا شده‌اند، اگرچه برخی از نقاط بین خوشه‌ها نزدیک به هم قرار دارند.

خوشه‌بندی با ۶ خوشه

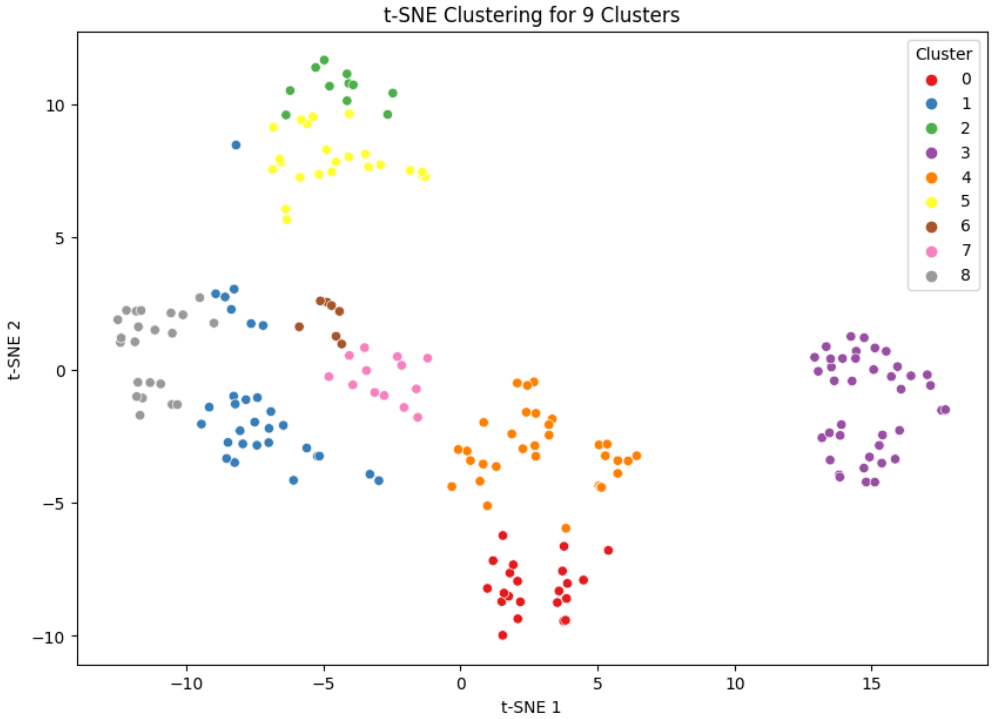


شکل ۱۸ خوشه‌بندی با ۶ خوشه

خوشه‌ها به خوبی از یکدیگر جدا شده‌اند و هر خوشه دارای تراکم خاص خود است.

تعداد بیشتری از خوشه‌ها ممکن است به تفکیک دقیق‌تر داده‌ها کمک کند.

خوشه‌بندی با ۹ خوشه



شکل ۱۹ نتایج خوشه‌بندی با ۹ خوشه

خوشه‌ها به وضوح از یکدیگر جدا شده‌اند و هر خوشه دارای ویژگی‌های منحصر به فرد خود است.

افزایش تعداد خوشه‌ها به تحلیل دقیق‌تر و شناسایی جزئیات بیشتر در داده‌ها کمک کرده است.

با توجه به نتایج حاصل از خوشه‌بندی با استفاده از t-SNE و الگوریتم K-means، به این نتیجه رسیدم که هر سه تعداد خوشه (۴، ۶ و ۹) به خوبی داده‌ها را به خوشه‌های متمایز تقسیم کرده‌اند. هر یک از این تعداد خوشه‌ها بسته به نیاز و دقت مورد انتظار می‌توانند انتخاب شوند:

۴ خوشه: برای تفکیک کلی و ساده‌تر داده‌ها مناسب است.

۶ خوشه: برای تحلیل دقیق‌تر و شناسایی ویژگی‌های بیشتر در داده‌ها مناسب است.

۹ خوشه: برای تحلیل جامع‌تر و شناسایی جزئیات بیشتر در داده‌ها مناسب است.

بررسی تأثیر ویژگی‌ها در t-SNE

۱. تحلیل حساسیت

در این روش، با تغییر جزئی در مقادیر هر ویژگی و مشاهده تغییرات در نمودار t-SNE، تأثیر آن ویژگی را بررسی می‌کنیم. اگر تغییرات کوچکی در ویژگی منجر به تغییرات بزرگی در نمودار شوند، می‌توان نتیجه گرفت که آن ویژگی تأثیر قابل توجهی دارد.

۲. حذف ویژگی‌ها

در این روش، هر ویژگی را به صورت جداگانه از داده‌ها حذف می‌کنیم و t-SNE را مجدداً اجرا می‌کنیم. سپس نتایج جدید را با نتایج اصلی مقایسه می‌کنیم. اگر حذف یک ویژگی خاص باعث تغییرات زیادی در نمودار t-SNE شود، می‌توان نتیجه گرفت که آن ویژگی اهمیت بالایی دارد.

۳. تحلیل بازسازی

این روش شامل استفاده از مدل‌های دیگر مانند PCA برای کاهش ابعاد و سپس بازسازی داده‌ها از ابعاد کاهش یافته است. سپس t-SNE را بر روی داده‌های بازسازی شده اجرا می‌کنیم و نتایج را با نتایج اصلی مقایسه می‌کنیم. این کار به ما کمک می‌کند تا بفهمیم که کدام ویژگی‌ها در کاهش ابعاد و خوشه‌بندی داده‌ها نقش مهمی دارند.

۸-۷_ جمع‌بندی

در این پروژه، از تکنیک‌های مختلفی برای کاهش ابعاد و خوشه‌بندی داده‌های مشتریان استفاده کردیم. هدف اصلی، تحلیل و بصری‌سازی داده‌ها برای درک بهتر از گروه‌بندی مشتریان و تأثیر ویژگی‌های مختلف بر خوشه‌بندی بود. تکنیک‌های مورد استفاده شامل PCA (Principal Component Analysis) و t-SNE (t-Distributed Stochastic Neighbor Embedding) بودند.

t-SNE در مقایسه با PCA، توانست خوشه‌های داده را به صورت بهتر و واضح‌تر نمایش دهد. این تکنیک با استفاده از رویکرد غیرخطی و احتمالاتی خود، ساختار واقعی و پیچیده داده‌ها را نمایش داد. با استفاده از t-SNE، توانستیم خوشه‌های جدا و متمایز را ببینیم که این امر به تحلیل بهتر و دقیق‌تر داده‌ها کمک کرد.

در نمودارهای t-SNE، با تغییر اندازه نقاط بر اساس ویژگی‌های مختلف، توانستیم تأثیر این ویژگی‌ها را بر خوشه‌بندی مشاهده کنیم. این تحلیل بصری به ما نشان داد که چگونه ویژگی‌هایی مانند جنسیت، سن،

درآمد و امتیاز خرید در هر خوشه توزیع شده‌اند و این اطلاعات می‌تواند به ما در تصمیم‌گیری‌های بازاریابی و استراتژی‌های کسب و کار کمک کند.

در نهایت، استفاده از t-SNE برای بصری‌سازی خوشه‌ها، درک بهتری از داده‌ها و خوشه‌بندی آن‌ها به ما داد. هرچند که این روش زمان بیشتری برای اجرا نیاز دارد و پارامترهای آن نیاز به تنظیم دقیق دارند، اما نتایج به دست آمده بسیار ارزشمند و قابل تفسیر بودند. همچنین، استفاده از PCA به عنوان یک روش سریع و کارا برای کاهش ابعاد اولیه و تحلیل خطی داده‌ها مفید بود. انتخاب تکنیک مناسب برای کاهش ابعاد و بصری‌سازی داده‌ها بستگی به نوع داده‌ها و اهداف تحلیل دارد و ترکیب این روش‌ها می‌تواند نتایج بهتری ارائه دهد.