

گزارش تکلیف ۵ درس یادگیری ماشین

کسرا سینایی

شماره دانشجویی ۸۱۰۶۹۶۲۵۴

۲۱ دی ۱۴۰۰

سؤال یک

الف

روش‌های جست و جو:

- Exhaustive: در این روش‌ها تمام حالات ممکن برای به دست آوردن زیرمجموعه‌ای از فیچرها امکان پذیر است در نظر گرفته می‌شوند. اگر n فیچر داشته باشیم، پیچیدگی محاسباتی این روش $O(n^2)$ است. به دلیل پیچیدگی محاسباتی بالا، این روش معمولاً کاربردهای کمی دارند (مثال: Breadth First Search)
- Heuristic: روش‌هایی مانند SFS, SBS, BDS و ... هستند. در این روش‌ها یا با مجموعه‌ی کامل فیچرها شروع کرده و به ترتیب فیچرهایی که حذف آن‌ها بهینه‌ترین زیرمجموعه جدید را نتیجه دهد حذف می‌شوند، یا با زیرمجموعه تهی از فیچرها شروع کرده و به مرور فیچرهایی را که بهینه‌ترین زیرمجموعه جدید را حاصل می‌کنند اضافه می‌شوند به زیرمجموعه فیچرها.
- Randomize: در این روش ابتدا به صورت اتفاقی زیرمجموعه‌ای از فیچرها انتخاب می‌شوند، سپس با استفاده از الگوریتم‌هایی مانند ژنتیک، RGSS و ... به بهینه سازی تابع هزینه می‌پردازند تا زیرمجموعه بهینه از فیچرها به دست آید.

روش‌های ارزیابی:

- Filter Methods: این روش‌ها بدون توجه به الگوریتم طبقه‌بندی زیرمجموعه انتخابی از فیچرها را ارزیابی می‌کنند. معیار اصلی ارزیابی اطلاعات موجود در هر زیرمجموعه از فیچر است. این روش‌ها سریع هستند و تمایل به انتخاب زیر مجموعه‌های بزرگی از فیچرها را داغرنند.
- Wrapper Methods: برای ارزیابی زیرمجموعه انتخاب شده از فیچرها، معیارهایی در نظر گرفته می‌شود که به الگوریتم طبقه بندی مربوط است. برای مثال پروسه ارزیابی کیفیت زیرمجموعه فیچر انتخاب شده از دقت آن در پیش بینی تعدادی داده تست استفاده می‌شود. این روش‌ها آهسته هستند اما دقیق تر از Filter Methods کار می‌کنند.

ب

در محاسبات LDA لازم است معکوس ماتریس S_w را حساب کرد و سپس مقادیر ویژه و بردارهای ویژه $S_w^{-1}S_b$ را به دست آورد. با افزایش ابعاد مسئله حجم محاسبات جبری افزایش می‌یابد. همچنین اگر تعداد نمونه‌ها کم باشد ممکن است ماتریس S_w سینگولار شود. در PCA نیز باید مقادیر ویژه ماتریس کواریانس نمونه‌ها و بردارهای متناظر با آن‌ها محاسبه شوند تا PCها به دست آیند. اگر ابعاد مسئله زیاد شود علاوه بر حجم محاسباتی مقادیر ویژه امکان کاهش دقت تخمین ماتریس کواریانس هم به وجود می‌آید. یکی از نقاط ضعف PCA حساسیت آن به اسکیل فیچرها می‌باشد به همین دلیل نرمال کردن دیتا قبل از اجرای الگوریتم اهمیت ویژه‌ای دارد.

سوال دو

$$\begin{aligned} \text{الف)} \quad S_T &= \sum_{\mathbf{x}} (\mathbf{x} - \mu)(\mathbf{x} - \mu)^T = \sum_{k=1}^C \sum_{\mathbf{x} \in D_k} (\mathbf{x} - \mu_k + \mu_k - \mu)(\mathbf{x} - \mu_k + \mu_k - \mu)^T \\ &= \underbrace{\sum_{k=1}^C \sum_{\mathbf{x} \in D_k} (\mathbf{x} - \mu_k)(\mathbf{x} - \mu_k)^T}_{S_W} + \underbrace{\sum_{k=1}^C \sum_{\mathbf{x} \in D_k} (\mu_k - \mu)(\mu_k - \mu)^T}_{S_B} \end{aligned}$$

$$\rightarrow S_T = S_W + S_B$$

ب) هر ترم از حاصل جمع های عبارت S_B به صورت quadratic است و از مرتبه 2

یک عبارت در transpose خود تشکیل می شود. بنابراین هر آلتر، رنگ این ماتریس ها

جزئی برابر یک می شود. در مسائل C کلاس، S_B حاصل جمع C تا از این ماتریس ها

است. پس رنگ آن هر آلتر می تواند C باشد. یک نکته باقی می ماند و آن، این است که

μ خود میانگین μ_k ها است و باعث می شود یک رجه آزادی از S_B سلب شود:

در نتیجه $\text{rank}(S_B) \leq C-1$ است.

سؤال چهار

$$P(x|w_i) \sim m_i, \Sigma_i$$

$$\text{cost: } J(w) = \frac{(\mu_1 - \mu_2)^2}{\delta_1^2 + \delta_2^2}$$

$$\text{projection: } y = w^T x \rightarrow P(y|w_i) \sim \mu_i, \delta_i^2$$

$$S_i = \sum_{x \in D_i} (x - m_i)(x - m_i)^T; S_W = S_1 + S_2^*$$

$$\begin{aligned} \mu_i &= w^T m_i; \delta_i^2 = \sum_{y \in D_i} (y - \mu_i)^2 = \sum_{y \in D_i} (w^T (x - m_i)) (w^T (x - m_i))^T \\ &= \sum_{x \in D_i} w^T (x - m_i)(x - m_i)^T w = w^T S_i w \end{aligned}$$

$$\Rightarrow \delta_1^2 + \delta_2^2 = w^T S_1 w + w^T S_2 w = w^T (S_1 + S_2) w = w^T S_W w \quad \textcircled{I}$$

$$\begin{aligned} S_B &= (m_1 - m_2)(m_1 - m_2)^T \quad (\mu_1 - \mu_2)^2 = (w^T m_1 - w^T m_2)^2 = w^T (m_1 - m_2)(m_1 - m_2)^T w \\ &= w^T S_B w \quad \textcircled{II} \end{aligned}$$

$$\text{I, II} \Rightarrow J(w) = \frac{(\mu_1 - \mu_2)^2}{\delta_1^2 + \delta_2^2} = \frac{w^T S_B w}{w^T S_W w}$$

$$\frac{\partial J}{\partial w} = \frac{(\frac{\partial}{\partial w} w^T S_B w) w^T S_W w - (\frac{\partial}{\partial w} w^T S_W w) w^T S_B w}{(w^T S_W w)^2} = 0$$

$$\rightarrow 2(S_B w) w^T S_W w - 2(S_W w) w^T S_B w = 0$$

$$\frac{(S_B w) w^T S_W w}{w^T S_W w} - \frac{(S_W w) w^T S_B w}{w^T S_W w} = 0 \rightarrow S_B w - \lambda S_W w = 0$$

$$S_B w = \lambda S_W w$$

$$\text{if } S_W \text{ is invertible} \rightarrow S_W^{-1} S_B w = \lambda w \quad \text{eigen value problem}$$

$$S_B w = (m_1 - m_2)(m_1 - m_2)^T w = \alpha (m_1 - m_2) \rightarrow S_B w \text{ is aligned with } m_1 - m_2$$

$$\Rightarrow w = S_W^{-1} (m_1 - m_2) = (\Sigma_1 + \Sigma_2)^{-1} (m_1 - m_2)$$

2000

سؤال پنج

الف) $\mu_1 = (1.33, 1.33)$ $\mu_2 = (2.5, 2)$ $m = (1.8, 1.6)$

$$S_B = \sum_{i=1}^2 N_i (\mu_i - m)(\mu_i - m)^T = 3 \begin{bmatrix} -0.47 \\ -0.27 \end{bmatrix} \begin{bmatrix} -0.47 & -0.27 \end{bmatrix} + 2 \begin{bmatrix} 0.7 \\ 0.4 \end{bmatrix} \begin{bmatrix} 0.7 & 0.4 \end{bmatrix}$$

$$\rightarrow S_B = \begin{bmatrix} 15.45 & 12.73 \\ 12.73 & 10.49 \end{bmatrix} + \begin{bmatrix} 2.42 & 2.64 \\ 2.64 & 2.88 \end{bmatrix} = \begin{bmatrix} 17.87 & 15.37 \\ 15.37 & 13.37 \end{bmatrix}$$

ب) $S_1 = \sum_x (x - \mu_1)(x - \mu_1)^T = \begin{bmatrix} -0.33 \\ -0.33 \end{bmatrix} \begin{bmatrix} -0.33 & -0.33 \end{bmatrix} + \begin{bmatrix} 0.66 \\ -0.33 \end{bmatrix} \begin{bmatrix} 0.66 & -0.33 \end{bmatrix} + \begin{bmatrix} -0.33 \\ 0.66 \end{bmatrix} \begin{bmatrix} -0.33 & 0.66 \end{bmatrix}$

$$\rightarrow S_1 = \begin{bmatrix} 0.1089 & 0.1089 \\ 0.1089 & 0.1089 \end{bmatrix} + \begin{bmatrix} 0.4356 & -0.2178 \\ -0.2178 & 0.1089 \end{bmatrix} + \begin{bmatrix} 0.1089 & -0.2178 \\ -0.2178 & 0.4356 \end{bmatrix} = \begin{bmatrix} 0.6534 & -0.3285 \\ -0.3285 & 0.6534 \end{bmatrix}$$

$$S_2 = \begin{bmatrix} -0.5 \\ 0 \end{bmatrix} \begin{bmatrix} -0.5 & 0 \end{bmatrix} + \begin{bmatrix} 0.5 \\ 0 \end{bmatrix} \begin{bmatrix} 0.5 & 0 \end{bmatrix} = \begin{bmatrix} 0.25 & 0 \\ 0 & 0 \end{bmatrix} + \begin{bmatrix} 0.25 & 0 \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} 0.5 & 0 \\ 0 & 0 \end{bmatrix}$$

$$S_W = S_1 + S_2 = \begin{bmatrix} 1.1534 & -0.3285 \\ -0.3285 & 0.6534 \end{bmatrix}$$

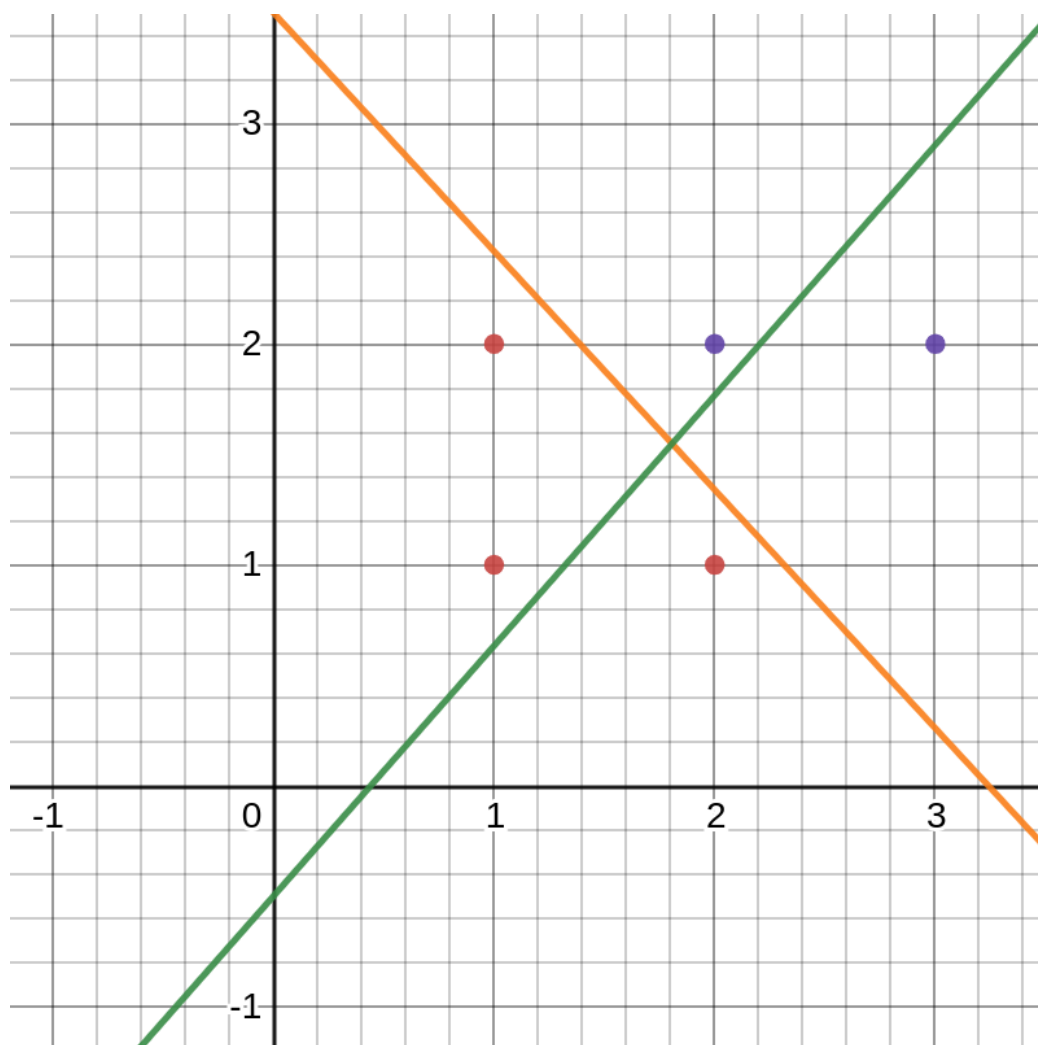
ج) $S_W^{-1} S_B = \begin{bmatrix} 1.011 & 0.508 \\ 0.508 & 1.786 \end{bmatrix} \begin{bmatrix} 17.87 & 15.37 \\ 15.37 & 13.37 \end{bmatrix} = \begin{bmatrix} 25.9 & 22.4 \\ 36.5 & 31.7 \end{bmatrix}$

eigen values: $\det \begin{vmatrix} 25.9 - \lambda & 22.4 \\ 36.5 & 31.7 - \lambda \end{vmatrix} = 0 \rightarrow (25.9 - \lambda)(31.7 - \lambda) - (22.4)(36.5) = 0$

eigen vectors: $\begin{bmatrix} -0.6544 \\ -0.5772 \end{bmatrix}$ $\begin{bmatrix} 0.756 \\ -0.8166 \end{bmatrix}$ $\lambda_1 = 0.0723$ $\lambda_2 = 57.53$

Best LDA directions: $\begin{cases} \text{tg}^{-1} \left(\frac{-0.5772}{-0.6544} \right) \\ \text{tg}^{-1} \left(\frac{-0.8166}{0.756} \right) \end{cases}$

با توجه به شیب خط‌های به دست آمده از قسمت قبل جهت‌های به دست آمده را همراه با داده‌ها رسم می‌کنیم.
نقاط قرمز مربوط به $y_i = 1$ و نقاط بنفش مربوط به کلاس $y_i = -1$ هستند.

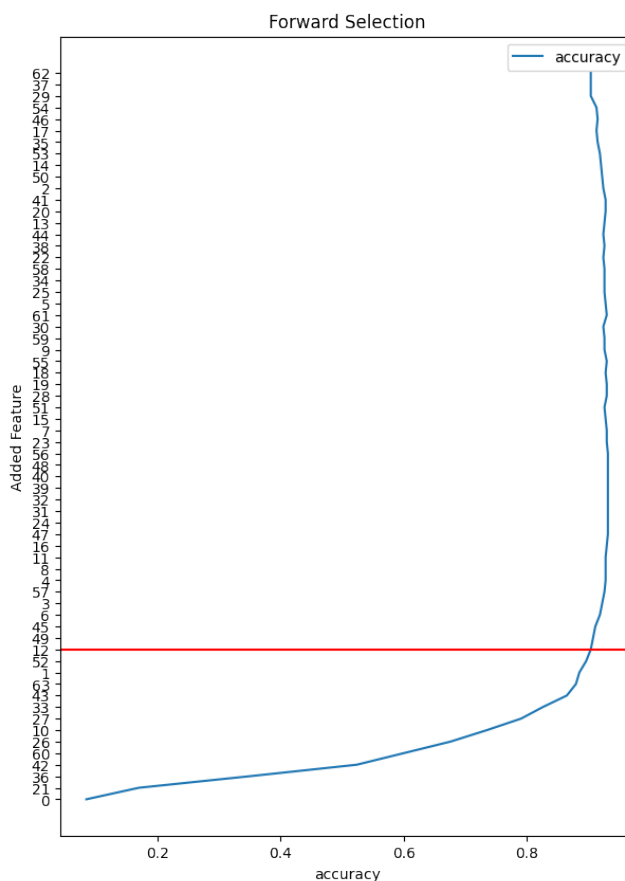


سؤال شش

الف

برای پیاده سازی SFS در حلقه for بیرونی دو متغیر تعریف می‌کنیم برای یافتن بهترین فیچر جهت اضافه کردن به لیست selected feature index. در حلقه for درونی که بر روی i لوپ می‌زند ابتدا چک می‌کنیم که i قبلاً انتخاب نشده باشد، سپس بردارهای x و x test را از روی متغیرهای کلاس و با توجه به ایندکس فیچرهای انتخاب شده تشکیل می‌دهیم و با استفاده از دستورات کتابخانه scikit learn دقت آن را بر روی مجموعه تست اندازه گیری می‌کنیم. پس از هر بار ایتريش بر روی متغیر i یک ایندکس به زیر مجموعه فیچرها اضافه می‌شود و دقت طبقه بندی آن نیز به لیست acc list اضافه می‌شود.

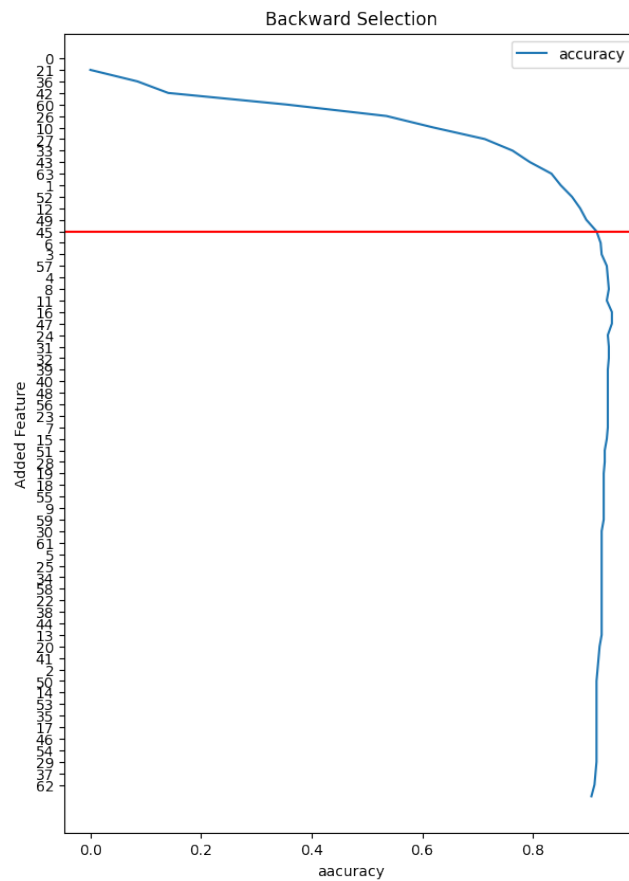
برای رسم نمودار مشابه نمودار مثال از کتابخانه pandas و matplotlib استفاده شده است. متغیر th (threshold) مقداری است که در آن دقت طبقه بندی با زیر مجموعه به دست آمده بیشتر از ۹۰٪ شود در نمودار با رنگ قرمز رسم شده است. تیک‌های محور y نیز لیست بازگردانده شده از متد forward می‌باشند. این نمودار در شکل ۱ آورده شده است.



شکل ۱: نتایج اجرای الگوریتم SFS

ب

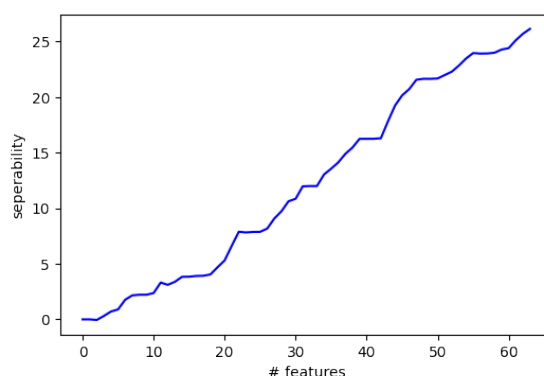
این قسمت شباهت زیادی به بخش الف دارد. فقط برای به دست آوردن زیرمجموعه فیچرها از تابع delete از کتابخانه numpy استفاده می‌کنیم تا بردارهای x و x test را بسازیم. نمودار رسم شده نیز مانند بخش قبل می‌باشد و تنها تفاوت در به دست آوردن مقدار th می‌باشد که باید از انتها به ابتدا لوپ زد و مقدار بهینه ترشهولد را به دست آورد. نمودار خواسته شده در شکل ۲ نشان داده شده است.



شکل ۲: نتایج اجرای الگوریتم SBS

سؤال هفت

برای تکمیل تابع Si که مقدار scatter matrix کلاس نام می باشد از رابطه $S_i = \sum_{x \in D_i} (x - m_i)(x - m_i)^T$ استفاده می کنیم. در این رابطه m_i میانگین داده های کلاس نام است. داده های ارسال شده در آرگومان ورودی این تابع باید مختص به یک کلاس باشند. جدا سازی داده های کلاس های مختلف در تابع Sb انجام می گیرد. در واقع از آرگومان دوم این تابع استفاده نشده است. برای پیاده سازی تابع Sw که within class scatter matrix می باشد از رابطه $S_w = \sum_{i=1}^c S_i$ استفاده می کنیم. آخرین تابع که برای محاسبه LDA به آن نیاز داریم Between class scatter matrix می باشد که در تابع Sb و به کمک رابطه $S_B = \sum_{i=1}^c N_i (m_i - m)(m_i - m)^T$ می توان آن را محاسبه کرد. در این رابطه m_i میانگین نمونه های کلاس نام و m میانگین داده های تمام کلاس ها است. N_i نیز تعداد نمونه های کلاس نام است. برای محاسبه ترنسفورمیشن های LDA مسئله مقادیر ویژه ارائه شده در کلاس درس را باید حل کنیم. بردار w بردار ویژه های ماتریس $S_W^{-1} S_B$ می باشد. برای اینکه تابع نوشته شده برای مواردی که ماتریس S_W سینگولار می شود نیز کار کند از pseudo inverse استفاده شده است. ورودی تمام کلاس ها باید آرایه numpy باشند. از روی دیتا ست استفاده شده می توان مقادیر data و labels را برای فراخوانی تمام توابع استفاده کرد. نمودار seperability بر حسب تعداد ویژگی ها نیز در شکل ۳ نشان داده شده است.



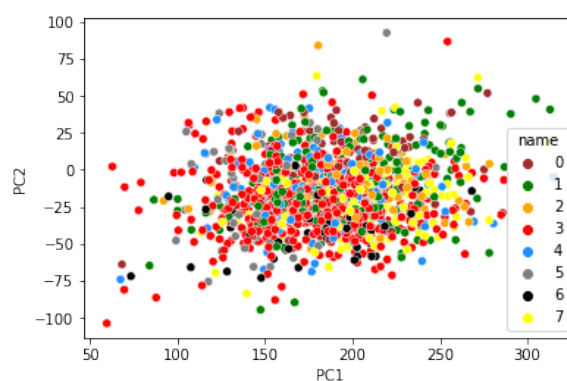
شکل ۳: معیار تفکیک پذیری بر حسب تعداد ویژگی های LDA

سؤال هشت

برای تکمیل تابع `PCATransform` کافی است با دستورات کتابخانه `numpy` ابتدا ماتریس کواریانس داده‌های ورودی را محاسبه کنیم و سپس بردارهای ویژه آن را بر حسب مقادیر ویژه متناظرشان سورت کنیم و در نهایت به تعداد `PC`های خواسته شده بازگردانیم. در این تابع علاوه بر `PC`های خواسته شده واریانس داده‌های تصویر شده بر روی بردارهای به دست آمده و مقادیر بردارهای ویژه که همان `PC`ها هستند نیز بازگردانده می‌شوند.

الف

در این قسمت با استفاده از خروجی اول تابع پیاده سازی شده که داده‌های تصویر شده بر روی `PC`ها هستند تصاویر چهره‌ها را بر روی دو کامپنت اول تصویر کرده و در نموداری مانند مثال رسم می‌کنیم. این نمودار در شکل ۴ نشان داده شده است.



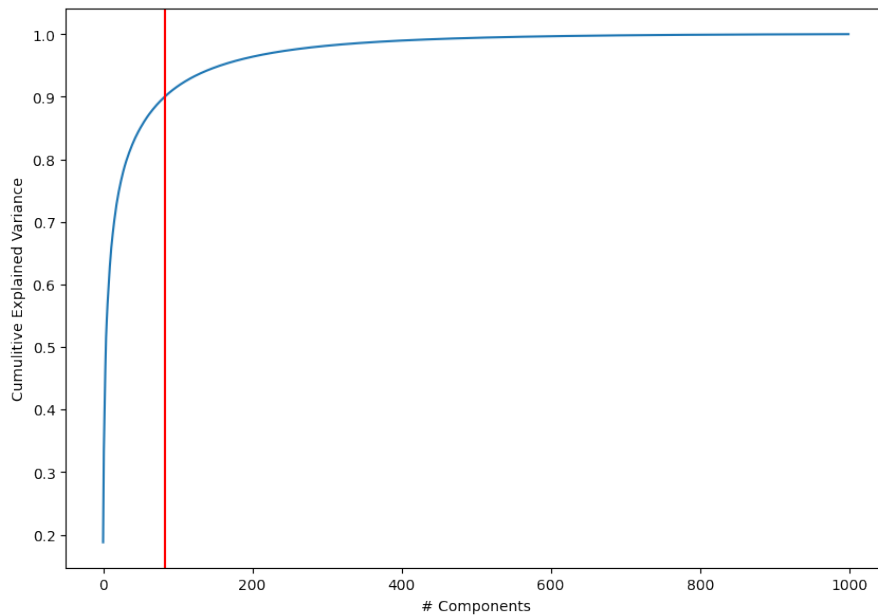
شکل ۴: تصاویر پس از تصویر شدن بر روی ۲ کامپنت اول PCA

ب

PCA یک الگوریتم `unsupervised` است که می‌توان به کمک آن جهت بردارهایی را پیدا کرد که اگر داده را بر روی آن‌ها تصویر کرد به داده‌های در ابعاد پایین تر رسید که بیشترین مقدار واریانس داده‌های اصلی را حفظ می‌کند. دو این جهت‌ها بردارهای ویژه ماتریس کواریانس هستند و دو کامپونت اول بردارهای ویژه‌ای هستند که اولین و دومین مقدار ویژه بزرگتر را نسبت به سایر بردارها دارند.

ج

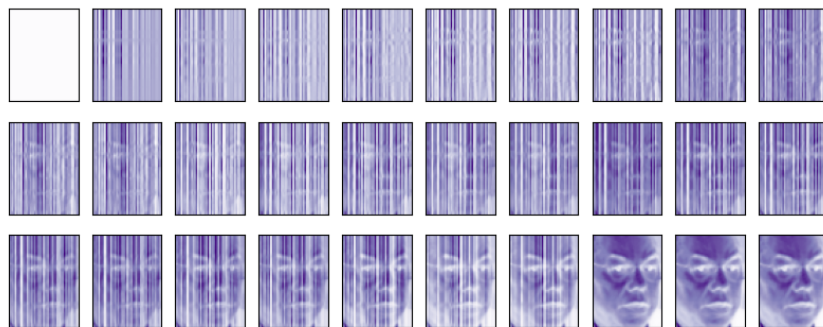
خروجی دوم تابع پیاده سازی شده در این سؤال نسبت واریانس بیان شده توسط داده‌های تصویر شده بر روی `n` کامپنت اول به صورت تجمعی را نشان می‌دهند. این آرایه `numpy` دقیقاً حاوی داده خواسته شده برای قسمت ب می‌باشد تنها لازم است مقدار ترشهولد را را نیز پیدا کنیم که به سادگی قابل یافتن می‌باشد. این مقدار بر روی نمودار با خط قرمز مشخص شده است. مقدار ترشهولد تعداد کامپونت‌هایی را نشان می‌دهد که با تصویرسازی داده‌ها بر روی آن‌ها می‌توان ۹۰٪ اطلاعات نمونه‌ها را پس از `dimensionality reduction` نشان داد.



شکل ۵: واریانس داده‌ها پس از انتقال بر روی PCها بر حسب تعداد کامپوننت

د

همانند تصاویر رنگی نشان داده شده در مثال داده‌ها را بر روی ۳۰ کامپوننت اول به دست آمده انتقال می‌دهیم و نتیجه تصویر باسازی شده از داده‌های انتقال داده شده در شکل ۶ نشان داده شده‌اند. با وجود اینکه تمام عکس‌های این سؤال



شکل ۶: ۳۰ کامپوننت اول

gray scale می‌باشند، در این بخش برای بهتر دیده شدن با مپ Blues نشان داده شده‌اند. قسمتهایی از عکس‌ها همان طور که در شکل ۶ نشان داده شده است روشن هستند و نوارهایی بر روی تصویر چهره به چشم می‌خورد. همان طور که می‌دانیم پس از تصویر سازی بر روی بردارهای PC قسمتی از داده‌ها (چه کم چه زیاد)

از بین می‌رود. این قسمت‌های سفید که در تصاویر با تعداد کامپوننت کم بیشتر نیز می‌باشند نشان دهنده قسمت‌های از دست رفته نمونه‌ها می‌باشد.

۵

در این قسمت به ازای مقدار ترشهولد به دست آمده در قسمت ج و همچنین دو مقدار کمتر از آن تصاویر هر هشت نفر انتقال داده شده و مجدداً با معکوس انتقال باز گردانده شده از تابع PCATransform عکس‌ها بازیابی شده‌اند. نتایج در شکل ۷ نشان داده شده‌اند.



شکل ۷: نتایج reconstruction تصاویر پس از انتقال به ازای تعداد مختلفی PC