Лабораторная работа №12. Метод главных компонент.

В лабораторной работе вы должны реализовать метод главных компонент (Principal component analysis – PCA) и посмотреть, как он работает для снижения размерности различных наборов данных. Теоретический материал для лабораторной работы можно найти в разделе 9 учебного пособия.

Сначала вы будете использовать набор данных размерности $\mathbb{R}^2$ для того, чтобы запрограммировать и проверить работу алгоритма. Затем полученные функции будут использованы для уменьшения размерности набора данных $\mathbb{R}^{1024}$ и подбора количества главных компонент для сохранения заданной дисперсии.

1 Загрузка, нормализация и отображение данных

На первом шаге программы загружается набор данных из файла ex7data1.npy. Далее необходимо выполнить нормализацию загруженных данных. Метод PCA работает только с нормализованными данными.

Найдите в программе функцию feature_normalize и дополните ее кодом. Подобные действия уже выполнялись в более ранних лабораторных работах. Функция принимает на вход матрицу X – исходную выборку. Матрица имеет размер $m \times n$, где m – число точек в выборке, n – размерность точек выборки. Функция должна вернуть нормализованную матрицу Xn того же размера, а также значения mu – среднего значения при нормализации и *sigma* – стандартного отклонения при нормализации.

Далее программе необходимо отобразить нормализованные данные. Найдите в программе функцию draw_data и дополните ее кодом для отображения точек выборки. Подобные действия уже выполнялись в более ранних работах. Входные параметры функции:

X – матрица исходных данных для отображения размером $m \times 2$, где m – число точек в выборке, каждая точка задана двумя координатами;

Xr – вторая такая же матрица для отображения, будет использована на третьем шаге работы, на данный момент можно ничего не делать с этой матрицей;

Функция не возвращает никаких значений.

В разделе, где вы должны добавить свой код выделено два подраздела: для шага 1 и для шага 3. На текущий момент в подразделе для шага 1 добавьте необходимые операторы для отображения исходной матрицы X. Вы должны получить график, похожий на рис. 1.

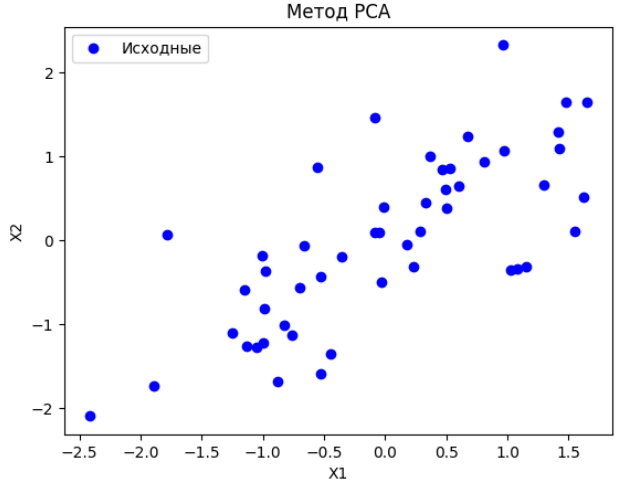

Рис. 1

Удалите оператор return, чтобы перейти к следующему шагу.

2 Метод главных компонент

На данном шаге вам необходимо реализовать метод главных компонент для снижения размерности данных от n к заданному числу k ($k < n$). Найдите в программе функцию pca и дополните ее необходимым кодом для реализации алгоритма. Функция принимает параметры:

X – матрица исходных данных размером $m \times n$, где m – число точек в выборке, n – размерность точек выборки;

k – требуемая размерность данных.

Функция должна вернуть:

Z – матрица размером $m \times k$ данных меньше размерности;

R – матрица понижения размерности для перехода от размерности $\mathbb{R}^n$ к $\mathbb{R}^k$ и наоборот.

Когда функция готова, запустите программу. Она вызывает написанную функцию для тестовых данных и выводит найденные значения. Если функция реализована правильно, полученные результаты должны совпадать с ожидаемыми.

Удалите оператор return, чтобы перейти к следующему шагу.

3 Восстановление данных

Используя полученную на предыдущем шаге матрицу понижения размерности R можно выполнить обратное восстановление данных от размерности $\mathbb{R}^k$ к $\mathbb{R}^n$, однако при этом часть информации теряется.

Найдите в программе функцию reconstruct и дополните ее необходимым кодом для восстановления размерности. Функция принимает параметры:

Z – матрица сжатых данных размером $m \times k$;

R – матрица понижения размерности.

Функция должна вернуть:

Xr – матрица приближенно восстановленных данных размером $m \times n$.

Для реализации алгоритма вам может понадобиться функция

```
U, S, V = np.linalg.svd(sigma)
```

Которая выполняет сингулярное разложение матрицы sigma. Подробное описание смотрите в документации (https://numpy.org/doc/stable/reference/generated/numpy.linalg.svd.html).

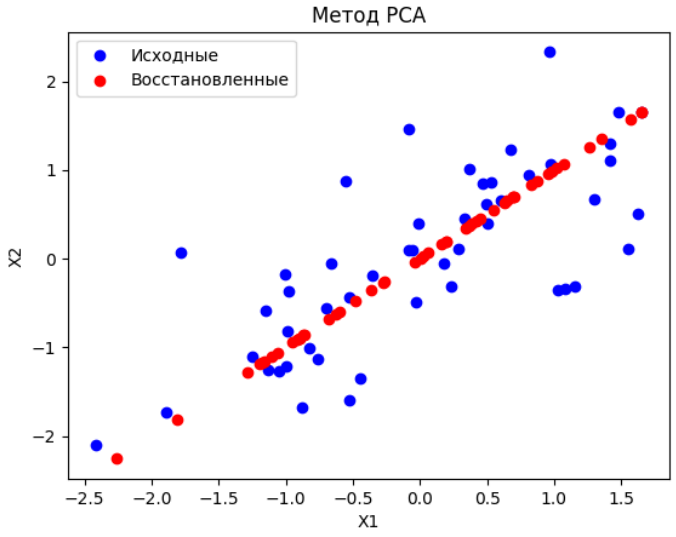

Рис. 2

Далее в программе отображаются полученные реконструированные после сжатия данные. Для этого найдите в программе функцию draw_data и дополните ее в разделе для шага 3 кодом отображения точек выборки Xr одновременно с точками X другим цветом. Если все выполнено правильно, вы должны получить график, похожий на рис. 2. Удалите оператор return, чтобы перейти к следующему шагу.

4 Применение PCA для набора данных лиц

На данном шаге загружается из файла ex7faces.npy второй набор данных. Он представляет собой 5000 полутоновых изображений лиц различных людей размером 32×32 пикселей. Матрица X, хранящая выборку, имеет размер 5000×1024, где каждая срока хранит изображение 32×32 (всего 1024 пикселей) в следующем виде: сначала 32 значения яркости первой строки изображения, затем второй и т.д. до последней строки.

После загрузки данных программа выбирает 25 случайных изображений из выборки и отображает их на экране. Для этого используется функция display_faces. Вам не нужно дополнять эту функция каким-либо кодом. Ознакомьтесь с ней и разберитесь с принципом ее работы.

Далее данные нормализуются с использованием вашей функции feature_normalize, понижаются от размерности $\mathbb{R}^{1024}$ к $\mathbb{R}^{100}$ с использованием вашей функции pca, а затем восстанавливаются в размерности $\mathbb{R}^{1024}$ с использованием вашей функции reconstruct. Полученные реконструированные данные отображаются на экране. Вы должны получить изображения, похожие на рис. 3.

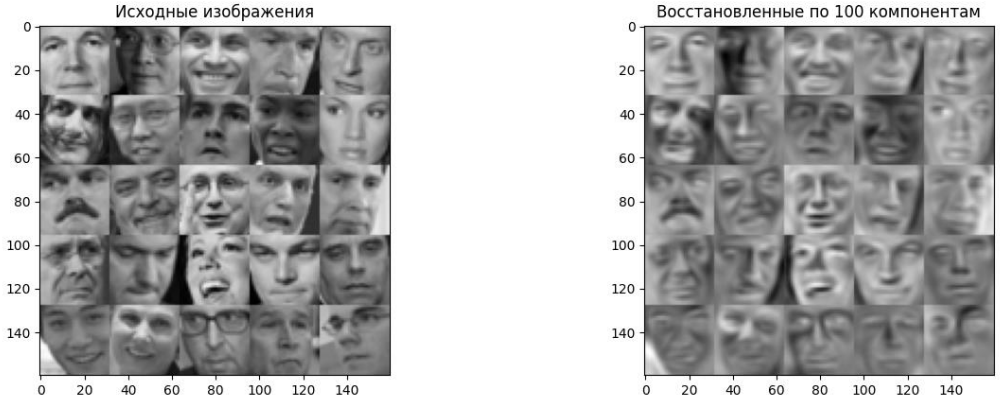

Рис. 3

Сравните эти изображения и сделайте вывод о восстановленных данных.

Удалите оператор return, чтобы перейти к следующему шагу.

5 Подбор значения *k*

Зачастую для алгоритма PCA заранее не известно число главных компонент, для которых необходимо снизить размерность, а известно, какую часть информации необходимо сохранить после понижения размерности, сколько процентов дисперсии данных должно сохраниться.

Найдите в программе функцию pca_adaptive и реализуйте в ней алгоритм pca с подбором значения k по требуемой величине остаточной дисперсии.

Функция принимает параметры:

X – матрица исходных данных размером $m \times n$;

$threshold$ – коэффициент требуемой минимальной величины сохраненной дисперсии данных, число в диапазоне 0..1, где 1 соответствует сохранению 100% всей

исходной дисперсии данных.

Функция должна вернуть:

Z – матрица размером $m \times k$ данных меньше размерности, где k – то значение числа главных компонент, при котором сохраняется не менее $threshold$ дисперсии;

R – матрица понижения размерности для перехода от размерности $\mathbb{R}^n$ к $\mathbb{R}^k$ и наоборот.

Обратите внимание, что функция np.linalg.svd, которую можно использовать для вычисления сингулярного разложения матрицы, вторым параметром S возвращает не диагональную матрицу собственных чисел (как показано в учебном пособии), а вектор одних только диагональных значений длиной n. Это сделано в библиотеке для сокращения объема используемой памяти, так как все остальные элементы матрицы всегда нулевые и не представляют интереса.

Когда функция готова, запустите программу. Функция pca_adaptive вызывается для понижения размерности исходных данных с сохранением 99% и 70% дисперсии. Если все реализовано правильно, вы должны получить 330-340 главных компонент для сохранения 99% дисперсии и порядка 10-15 компонент для сохранения 70% дисперсии.

Также программа выводит изображения восстановленных после сжатия данных. Сравните качество изображений и сделайте выводы.

На этом выполнение лабораторной работы №12 завершается.