НЕТОЛОГИЯ

Юстина Иванова

Программист, data scientist

Кейс-стади

НЕТОЛОГИЯ

Спикер

Юстина Иванова,
Data scientist по
Компьютерному зрению
в компании Dataplex
Выпускница МГТУ им. Баумана
Msc Artificial Intelligence,
University of Southampton

Основные непонятные моменты

1. Матрица ковариаций
2. Свойства матрицы ковариаций
3. Смысл ковариационной матрицы
4. Проецирование данных на вектор
5. Скалярное произведение.
6. Транспонирование матрицы
7. Собственные векторы
8. Собственные значения
9. Спектральное разложение матрицы
10. Разложение Холецкого
11. Декомпозиция матрицы ковариаций
12. Матрица преобразований
13. Теория вероятности
14. Условная вероятность.

Основные понятия

Дисперсия — среднеквадратичное отклонение от среднего значения (насколько данные разбросаны)

$$\sigma^2(x) = \frac{1}{n}\sum_{i=1}^{n}(x_i - \mu)^2$$

Ковариация — наличие зависимости между величинами

$$\sigma(x,y) = \frac{1}{n}\sum_{i=1}^{n}(x - \mu_x)(y - \mu_y)$$

Корелляция — нормированная ковариация, определяет силу зависимости

$$\sigma(x,y) = \frac{Cov(x,y)}{\sqrt{Var(x)}\sqrt{Var(y)}} = \frac{\frac{1}{n}\sum_{i=1}^{n}(x - \mu_x)(y - \mu_y)}{\sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - \mu_x)^2}\sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \mu_y)^2}}$$

Матрица ковариаций

Матрица корелляций подсчитывается с помощью формул, которые показывают как данные зависят друг от друга в пространстве n значений (каждый элемент матрицы равен ковариации двух выборок).

$$\mathbf{\Sigma} = \begin{bmatrix} \sigma(X_1, X_1) & \sigma(X_1, X_2) & \dots & \sigma(X_1, X_n) \\ \sigma(X_2, X_1) & \sigma(X_2, X_2) & \dots & \sigma(X_2, X_n) \\ \vdots & \vdots & \ddots & \vdots \\ \sigma(X_n, X_1) & \sigma(X_n, X_2) & \dots & \sigma(X_n, X_n) \end{bmatrix}$$

Смысл матрицы ковариаций

Большинство решаемых проблем в data science и машинном обучении идет через нахождение матрицы ковариаций:

- Классификационный анализ
- Регрессионный анализ
- Метод главных компонент
- Метод Фишера

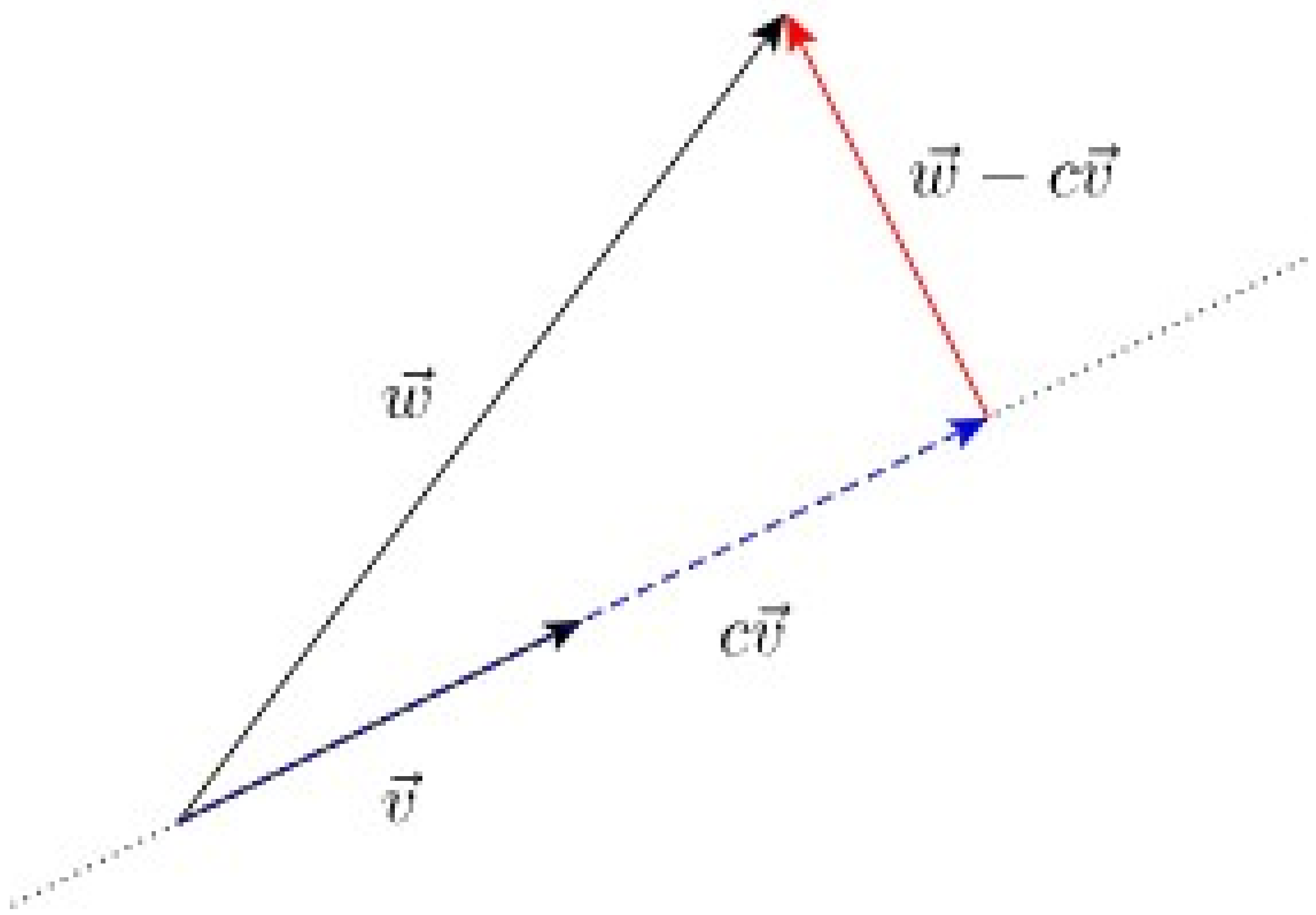
Классификация

Проблема классификации может решатся такими методами как:

- Дерево решений
- Наивная байесовская классификация
- Метод наименьших квадратов
- Метод опорных векторов

Проецирование данных на вектор

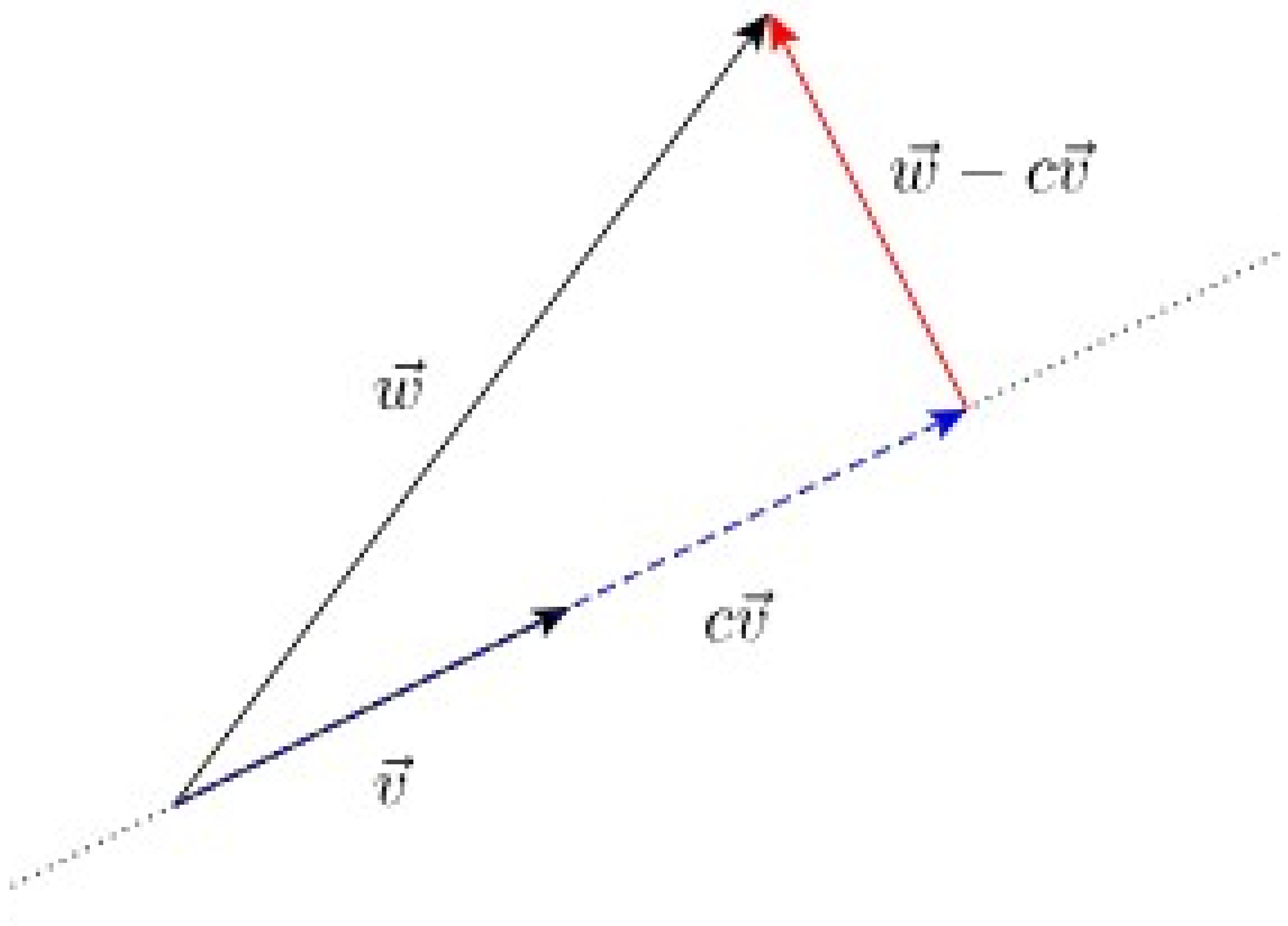
Чтобы посчитать расстояние между точкой и прямой, необходимо знать как проецировать вектор на прямую.



cv = np.dot(**w, v**)/np.dot(**v,v**)***v**.

http://qaru.site/questions/15739903/project-vector-w-onto-vector-v-and-draw-perpendicular-line-preparation-for-pca

Скалярное произведение



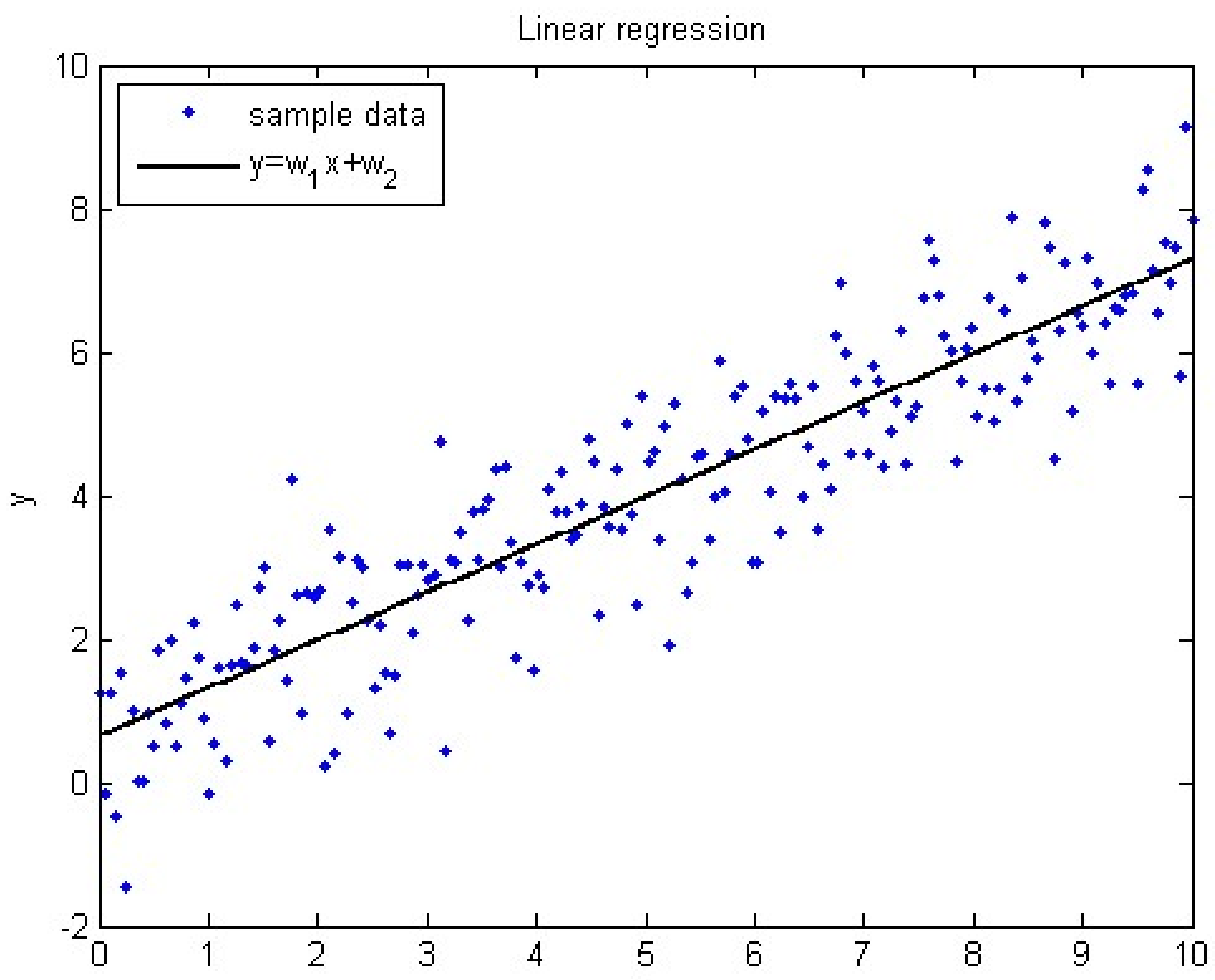
Необходимо для выполнения проецирования данных на вектор.

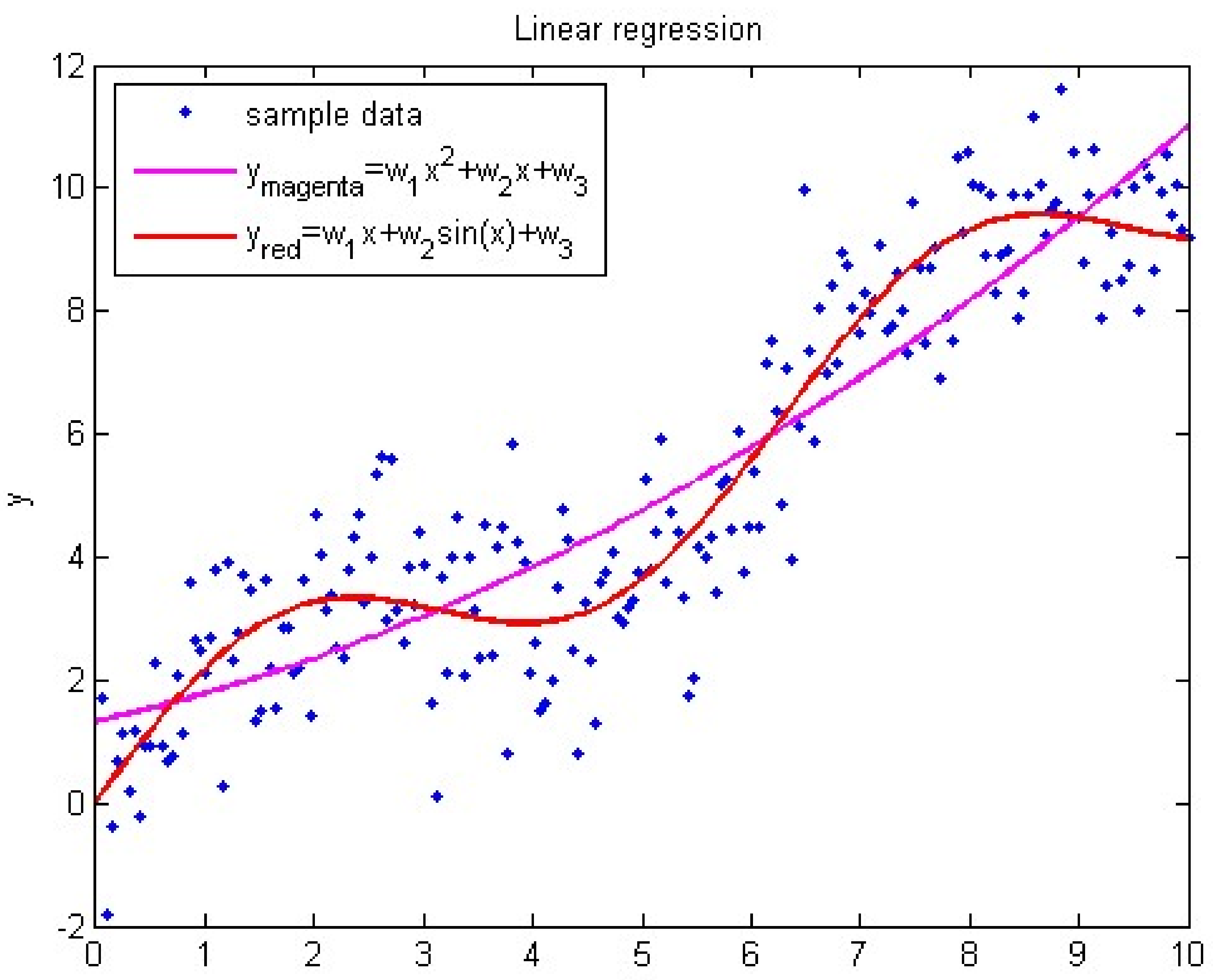
np.dot(w,v) = w1*w2 + v1*v2
Где w = (w1, w2)
V = (v1, v2)

cv = np.dot(**w, v**)/np.dot(**v,v**)***v**.

http://qaru.site/questions/15739903/project-vector-w-onto-vector-v-and-draw-perpendicular-line-preparation-for-pca

Линейная регрессия





Для заданного пространства данных найти уравнение прямой (или кривой), равновесно разделяющее данные посередине.

http://www.machinelearning.ru/wiki/index.php?title=%D0%A0%D0%B5%D0%B3%D1%80%D0%B5%D1%81%D1%81%D0%B8%D0%BE%D0%BD%D0%BD%D1%8B%D0%B9_%D0%B0%D0%BD%D0%B0%D0%BB%D0%B8%D0%B7

Транспонирование матрицы

Первая строка стала первым столбцом

$$A = \begin{pmatrix} \boxed{-1 \quad 2 \quad 4 \quad 0 \quad 7} \\ \boxed{3 \quad -5 \quad 24 \quad 9 \quad -3} \\ \boxed{-10 \quad -8 \quad -2 \quad -4 \quad 11} \end{pmatrix} \qquad A^T = \begin{pmatrix} -1 & 3 & -10 \\ 2 & -5 & -8 \\ 4 & 24 & -2 \\ 0 & 9 & -4 \\ 7 & -3 & 11 \end{pmatrix}$$

Вторая строка стала вторым столбцом

Третья строка стала третьим столбцом

Транспонирование матрицы в python:

```
import numpy as np
a = np.array([[0, 1, 2], [4, 5, 6]])
a = a.transpose()
или
a = a.T
```

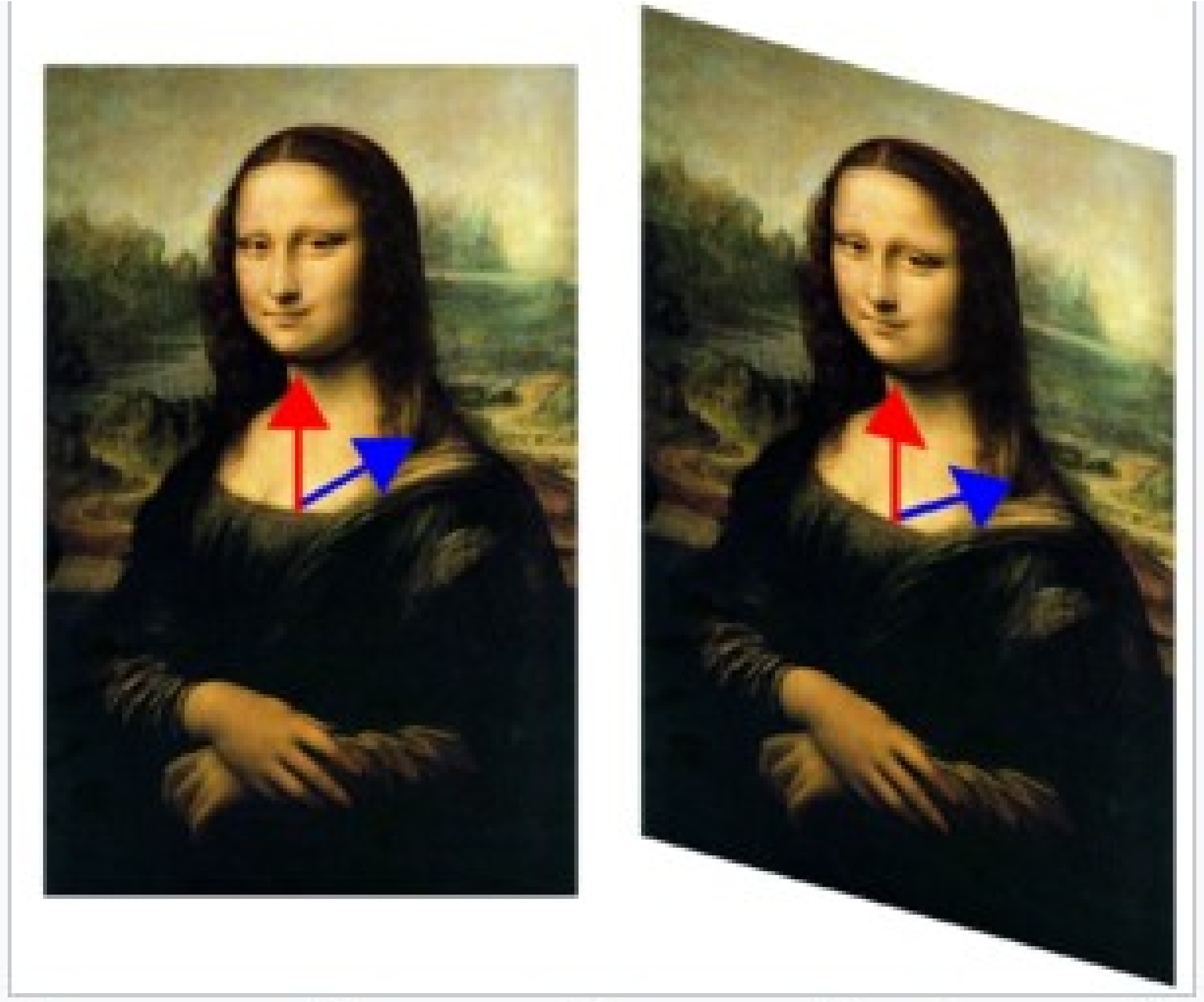
https://math1.ru/education/matrix/matrixop.html

Собственные вектора и собственные значение

Считаются через ковариационную матрицу.

Собственным вектором линейного преобразования **A** называется такой ненулевой вектор **x** в **L**, что для некоторого **lambda** в **K**, где **L** — линейной пространство над полем **K**

Ax=lambda x



Другая трансформация Джоконды. Синий вектор меняет направление, а красный — нет. Поэтому красный является собственным вектором, а синий — нет. Так как красный вектор ни растянулся, ни сжался, его собственное значение равно, как и на картинке выше, единице. Все векторы, коллинеарные красному, тоже собственные.

https://ru.wikipedia.org/wiki/%D0%A1%D0%BE%D0%B1%D1%81%D1%82%D0%B2%D0%B5%D0%BD%D0%BD%D1%8B%D0%B9_%D0%B2%D0%B5%D0%BA%D1%82%D0%BE%D1%80

Декомпозиция матрицы ковариаций

Это нахождение собственных векторов
и собственных значений.

Scipy и Numpy имеют между собой три различные функции для поиска собственных векторов для заданной квадратной матрицы:

numpy.linalg.eig(a)
scipy.linalg.eig(a) и
scipy.sparse.linalg.eig(A, k)

Вопросы от участников

Подскажите мне пожалуйста

import numpy.linalg as la

n = 1000

C = [[1,0.98],[0.98,1]] - **Это я так понял матрица ковариации. Почему такие цифры - просто для примера?**

A = la.cholesky(C)

X = np.random.randn(n,2) - **что это? - генерит 2 рандомных числа от одного до 1000? Ваша фраза X на самом деле выглядит как [x, y] -не понятно**

Y = np.dot(A,X.T) - **Умножаем нашу матрицу на чтото сверху? что такоеX.T?**

plt.plot(X[:,0], X[:,1], 'r.') - **мне непонятно что это такое X[:,0], X[:,1]** просьба обьяснить

plt.plot(Y[0,:], Y[1,:], 'b.') (edited)

Ссылки на кейс-стади

https://github.com/yustiks/statistics_in_python/blob/master/statistics_4.ipynb

Контакты спикера

yustiks@gmail.com