Lecture 15: Mathematics of Data Science Algorithms: Matrix Algebra and Calculus

Examples: Principal Component Analysis and Gradient Descent

Heidi Perry, PhD

Hack University

heidiperryphd@gmail.com

12/05/2016

Overview

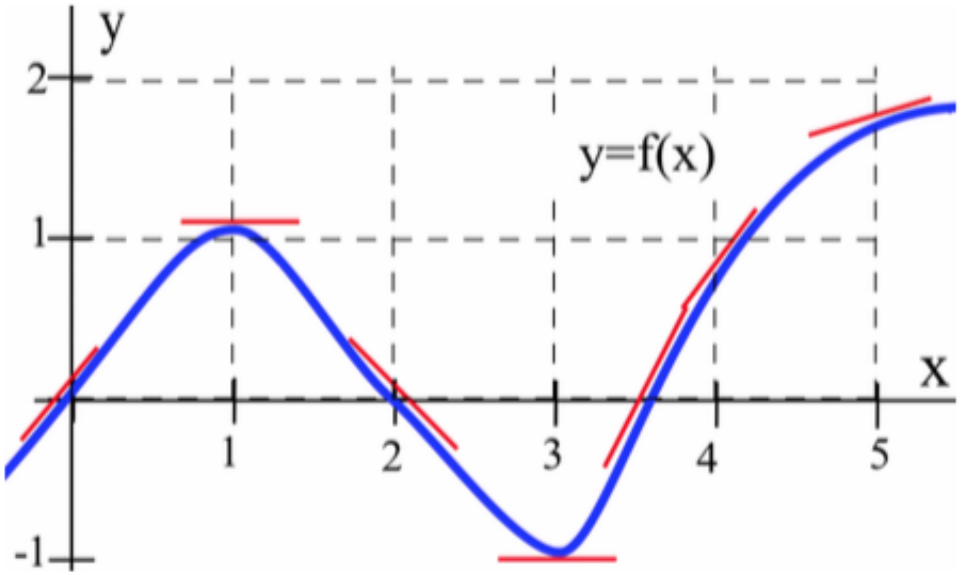
1. Calculus
   - Derivatives
   - Gradient Descent
2. Linear Algebra
   - Introduction to Vectors
   - Introduction to Matrices
   - Change of Basis

Descent Algorithm

Imagine you are a (very intelligent) mouse trying to make it to that lake. You can only see the land maybe a foot around you. How would you proceed?

Picture from Laurie at 59 The View From Here

Derivative is the slope of the tangent line



[Hoffman, 2016]

x	$y = f(x)$	$m(x)$
0	0	1
1	1	0
2	0	-1
3	-1	0
4	1	1
5	2	$\frac{1}{2}$

where $m(x)$ is the estimated **slope** of the tangent line to the graph of f(x) at the point (x, y)

Derivative Basics

Definition of the Derivative

$$\frac{df}{dx} = \lim_{h \to 0} \frac{f(x+h) - f(x)}{h}$$

Derivative Notation

- $f'(x)$ emphasizes that the derivative is a function related to the function f
- $\mathbf{D}(f)$ emphasizes that f' is a result of an operation on f
- $\frac{df}{dx}$ emphasizes that the derivative is the limit of $\frac{\Delta f}{\Delta x} = \frac{f(x+h)-f(x)}{h}$

[Hoffman, 2016]

Properties of a Function Related to the Derivative

Tangent Line Formula

If $f(x)$ is differentiable at $x = a$ then an equation of the line tangent to f at $(a, f(a))$ is:

$$y = f(a) + f'(a)(x - a)$$

- $f(x)$ is **increasing** if the value increases as the input x move from left to right. $\frac{df}{dx} > 0$
- $f(x)$ is **decreasing** if the value decreases as the input x move from left to right. $\frac{df}{dx} < 0$
- Points where $\frac{df}{dx} = 0$ are **critical points**: (local) maximum, (local) minimum, or inflection point.

Properties of the derivative

- **Theorem:** If a function is differentiable at a point, then it is continuous at that point.
- **Contrapositive form of previous theorem:** If f is not continuous at a point, then f is not differentiable at that point.
- Other times a function is not differentiable:
  - At a cusp or corner.
  - When the tangent line is vertical.

Optimization: Numerical Root-finding Methods

Find minimum (or maximum) of a function by looking for **roots** (points where the function value is zero) of the derivative.

- Bisection
  1. Find interval $[a, b]$ such that $f(a)$ and $f(b)$ have opposite signs.
  2. Bisect interval, $m = \frac{a+b}{2}$.
  3. If $f(m) = 0$, return m. If $f(m) \neq 0$, set $a = m$ or $b = m$ such that condition (1) is met. Repeat until $f(m) < \epsilon$ where ϵ is a threshold that is "close-enough" to zero.
- Newton's method
  1. Pick a starting value x_0.
  2. For each x_n, calculate a new estimate $x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$ (this "steps" in the direction the tangent line "points" to).
  3. Repeat step 2 until the estimates are "close enough" to a root or until the method fails.

[Hoffman, 2016]

Gradient: Multi-dimensional derivative

Let f be a function of many variables: $f(x_1, x_2, x_3, ..., x_n) = f(\vec{x})$
A partial derivative with respect to any one of the variables is defined as the derivative of the function with all other variables held constant:

$$\frac{\partial f(\vec{x})}{\partial x_i} = \frac{df(x_i)}{dx_i} \text{ where } x_j \neq x_i \text{ are treated as constants}$$

The gradient operator is a vector of partial derivative operators:

$$\nabla = \begin{pmatrix} \frac{\partial}{\partial x_1} \\ \frac{\partial}{\partial x_2} \\ \vdots \\ \frac{\partial}{\partial x_n} \end{pmatrix} \text{ so } \nabla f(\vec{x}) = \begin{pmatrix} \frac{\partial f(\vec{x})}{\partial x_1} \\ \frac{\partial f(\vec{x})}{\partial x_2} \\ \vdots \\ \frac{\partial f(\vec{x})}{\partial x_n} \end{pmatrix}$$

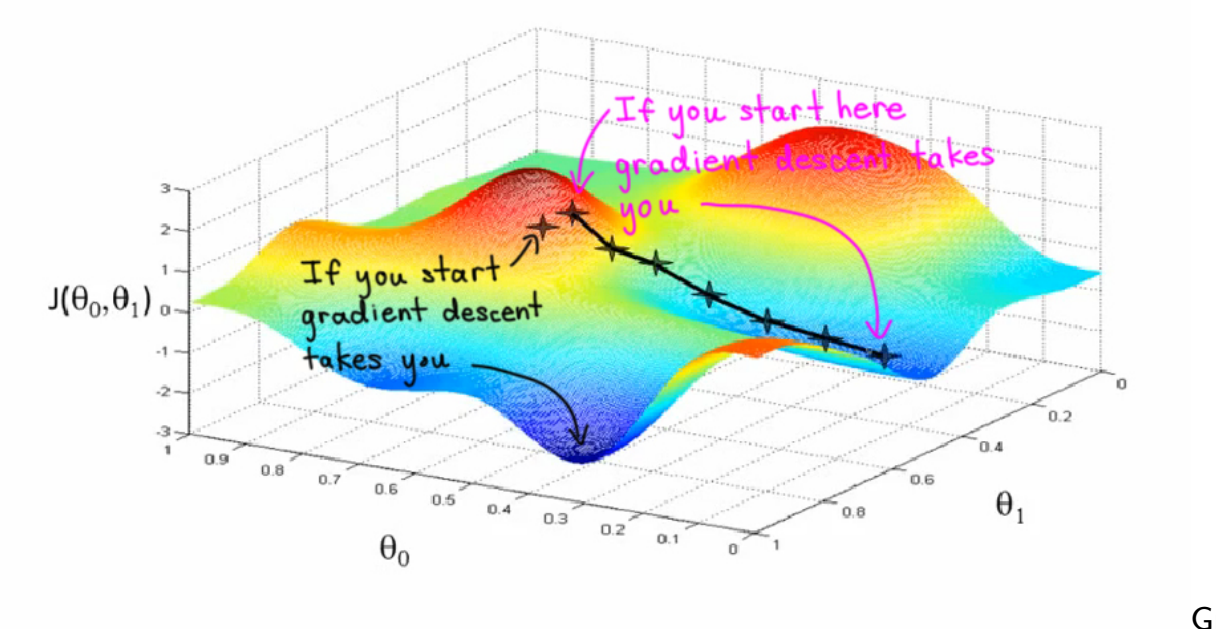
The gradient vector points in the direction of greatest **increase** of the function.

Gradient Descent

Gradient descent algorithm

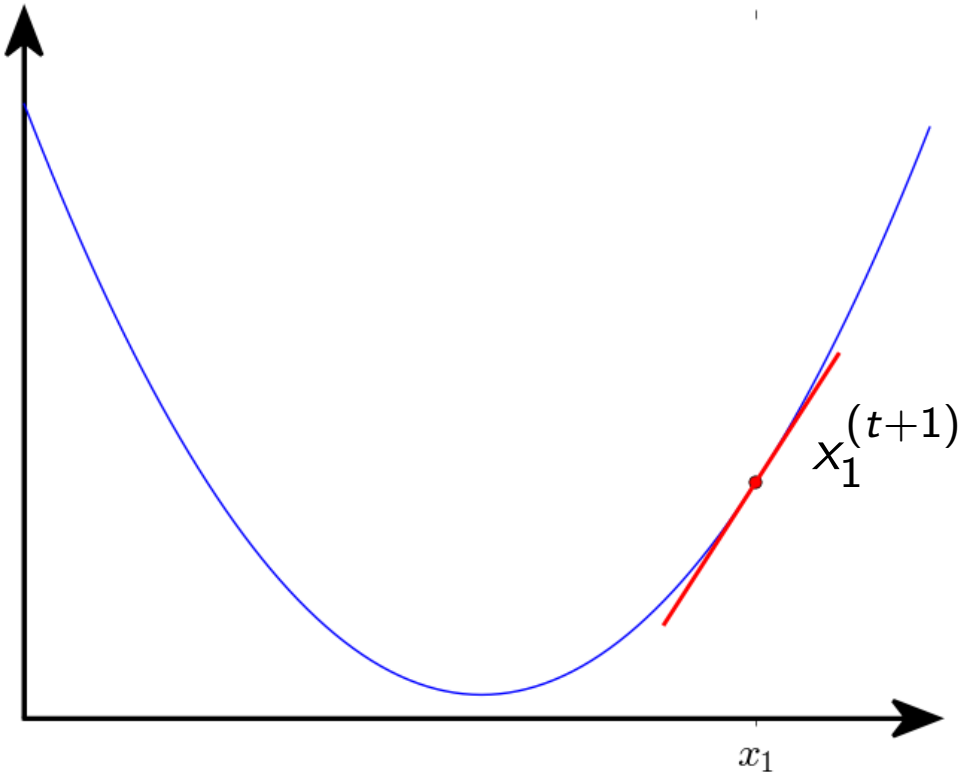
repeat until convergence:

$$\vec{x} := \vec{x} - \alpha \nabla f(\vec{x})$$



Graphic from quinnliu

One-dimensional Gradient Descent: Derivative Term



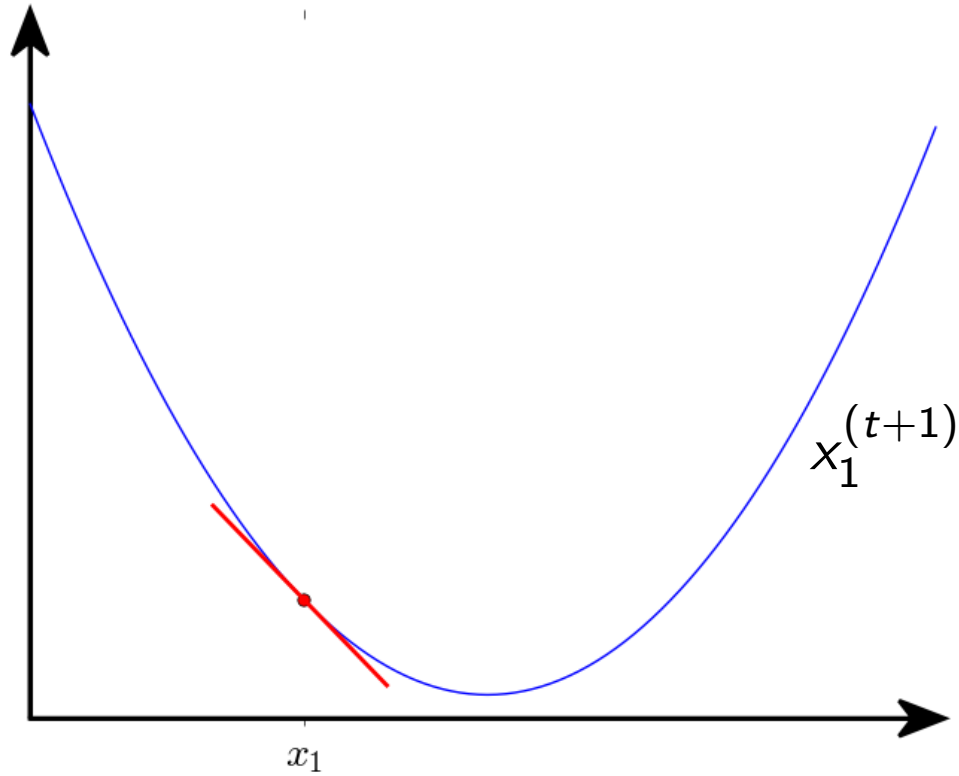
$$x_1^{(t+1)} = x_1^{(t)} - \alpha \frac{d}{dx} f(x)$$

$$\frac{d}{dx} f(x) > 0$$

$$x_1^{(t+1)} = x_1^{(t)} - \alpha * (\text{positive number})$$

x_i decreases

One-dimensional Gradient Descent: Derivative Term



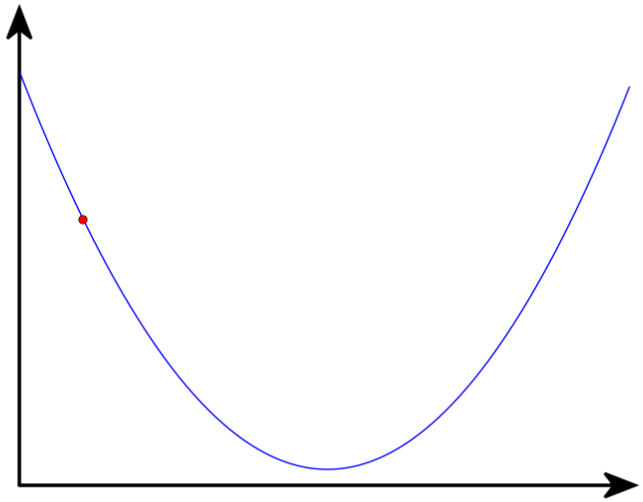
$$x_1^{(t+1)} = x_1^{(t)} - \alpha \frac{d}{dx} f(x)$$

$$\frac{d}{dx} f(x) < 0$$

$$x_1^{(t+1)} = x_1^{(t)} - \alpha * (\text{negative number})$$

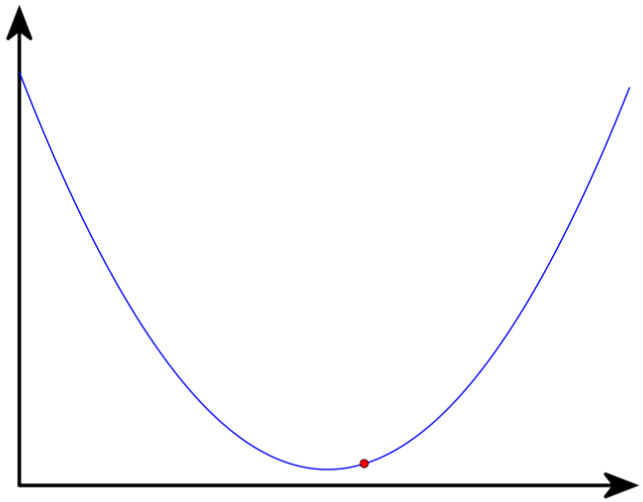
x_1 increases

One-dimensional Gradient Descent: Alpha Term



$$x_1^{(t+1)} = x_1^{(t)} - \alpha \frac{d}{dx} f(x)$$

if α is too small, gradient descent is slow



if α is too large, gradient descent will overshoot the minimum and fail to converge, or even diverge

Apply Gradient Descent Algorithm to Linear Regression

Gradient descent algorithm

repeat until convergence:

$$w_j^{(t+1)} = w_j^{(t)} - \alpha \frac{\partial J}{\partial w_j}\bigg|_{w^{(t)}}$$

for $j = 1$ and $j = 0$

Linear Regression Model

$$h_\theta(x) = \theta_0 + \theta_1 x$$

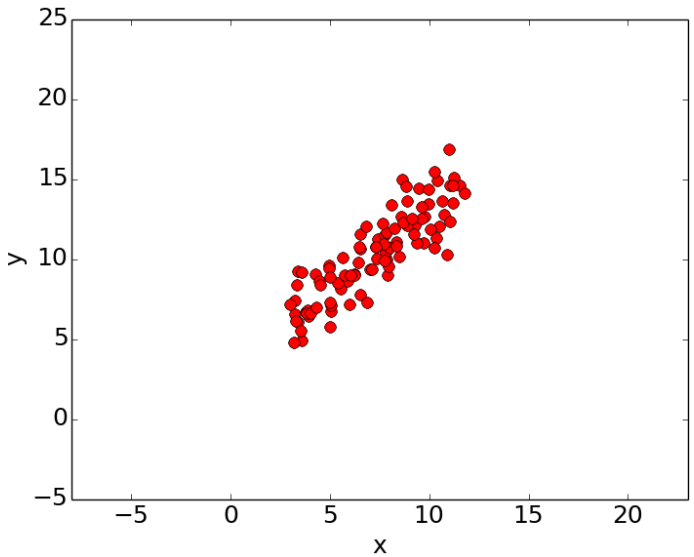
$$J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)}\right)^2$$

where $x^{(i)}$, $y^{(i)}$ are the observations of the explanatory and response variables respectively

Gradient Descent for Linear Regression Example

The following example, including all graphics, is from Matt Nedrich. Code available on GitHub.

Scatter Plot of Data



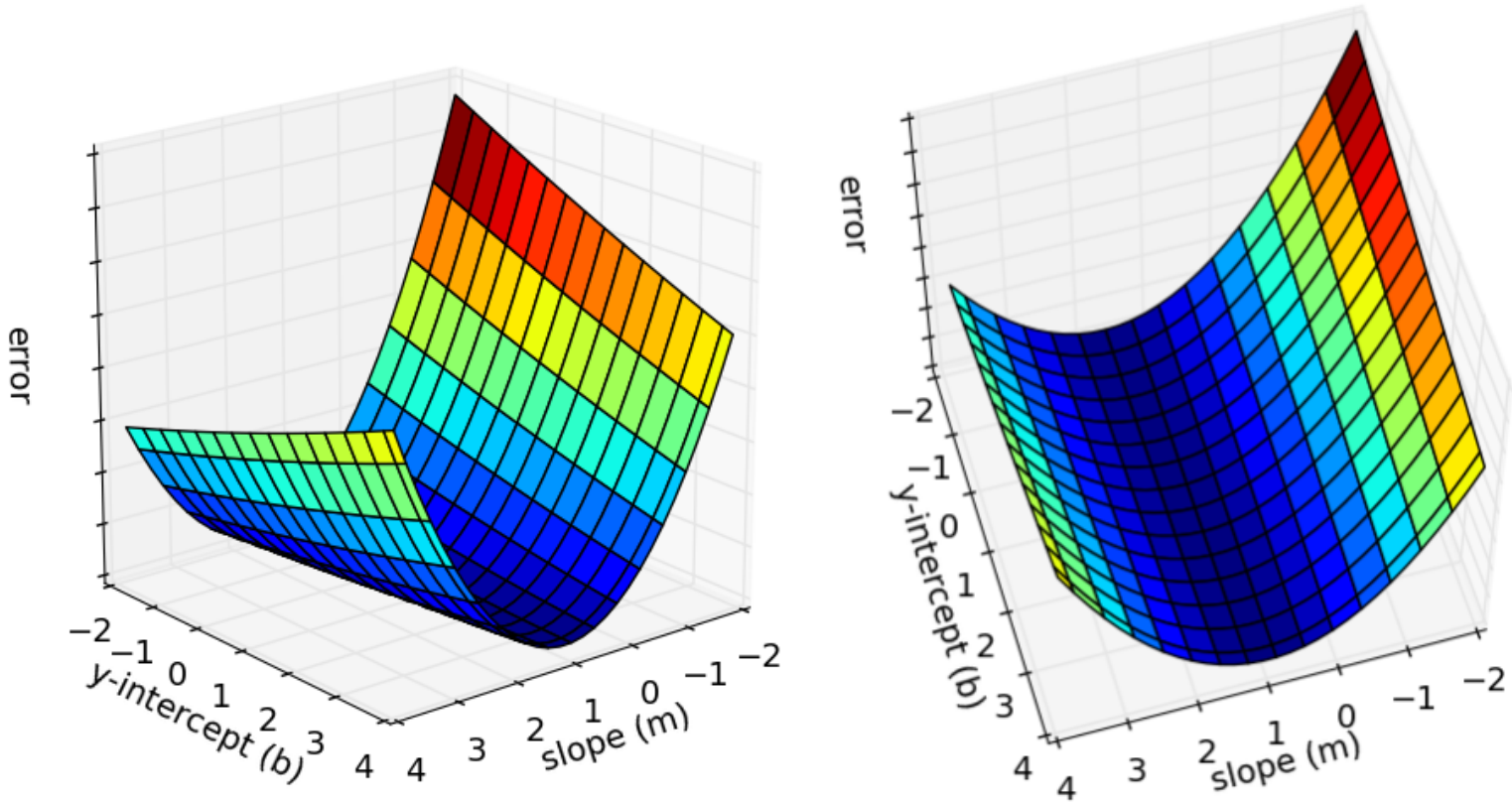
Error term to minimize:

$$J(m,b) = \frac{1}{N}\sum_{i=1}^{N}(y_i - (mx_i + b))^2$$

Gradient:

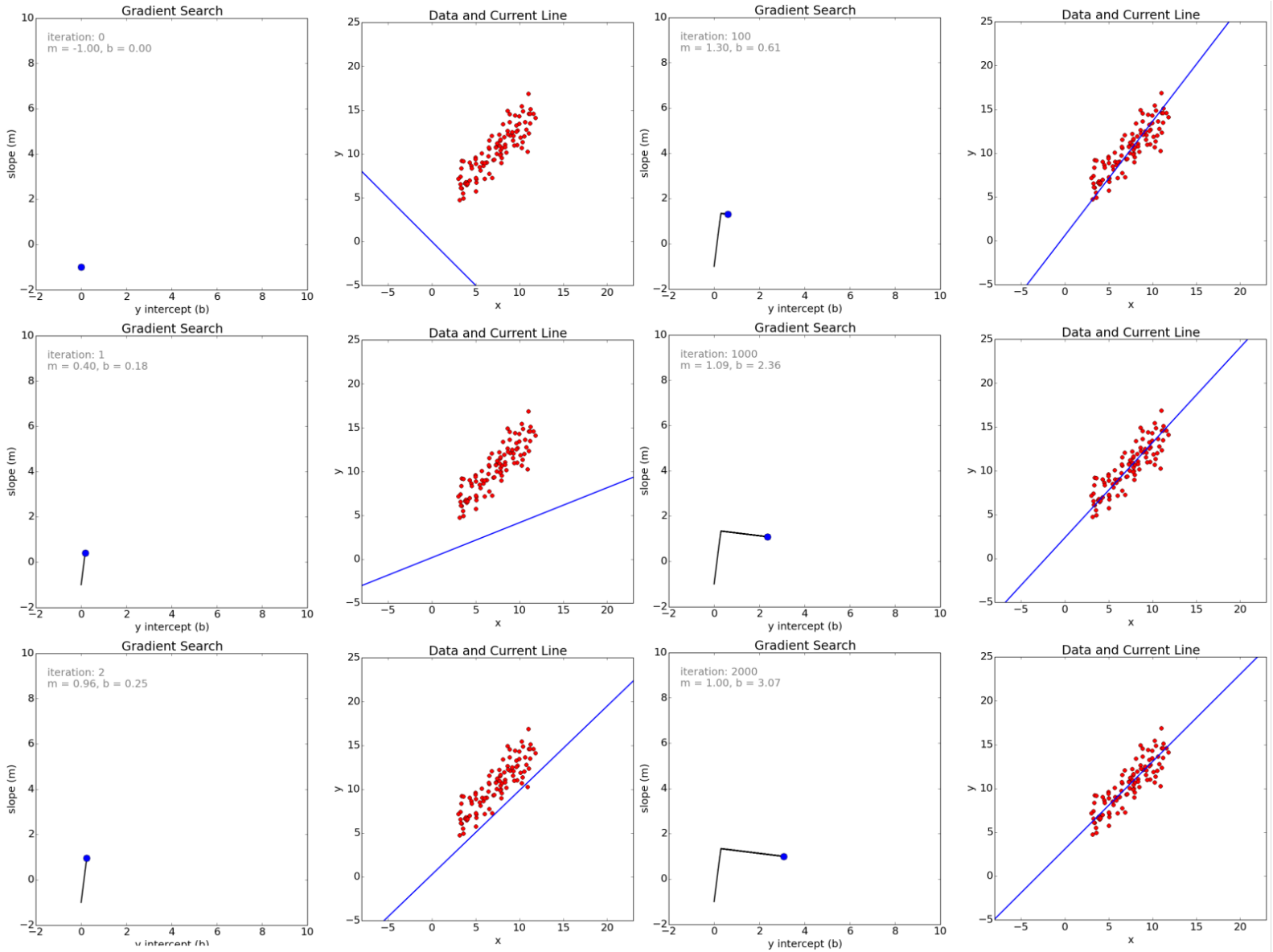
$$\nabla J = \begin{bmatrix} \frac{\partial J}{\partial b} \\ \frac{\partial J}{\partial m} \end{bmatrix} = \begin{bmatrix} \frac{2}{N}\sum_{i=1}^{N} -(y_i(mx_i + b)) \\ \frac{2}{N}\sum_{i=1}^{N} -x_i(y_i(mx_i + b)) \end{bmatrix}$$

Gradient Descent for Linear Regression Example

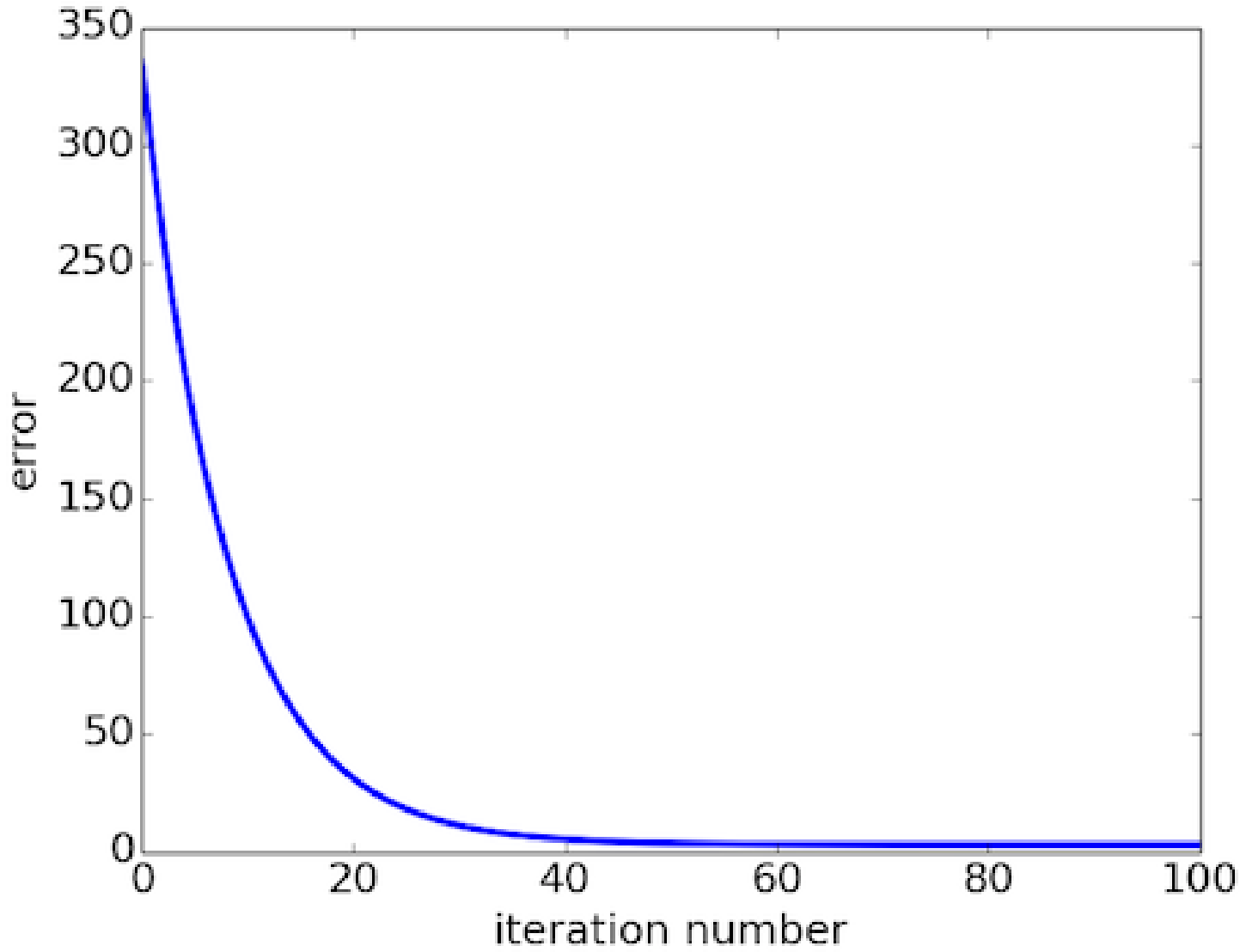


Start gradient search at point $(b, m) = (0, -1)$

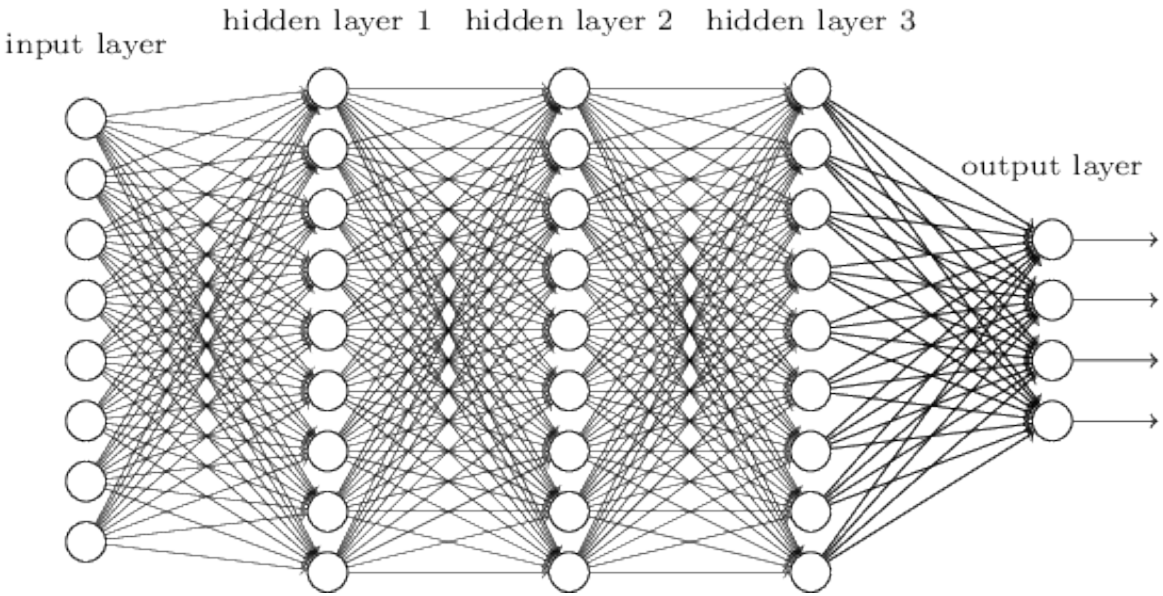
Gradient Descent for Linear Regression Example



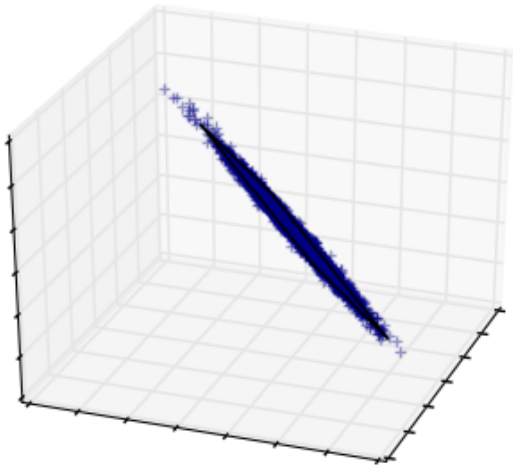
Gradient Descent for Linear Regression Example



Neural Nets and Reducing Dimensionality



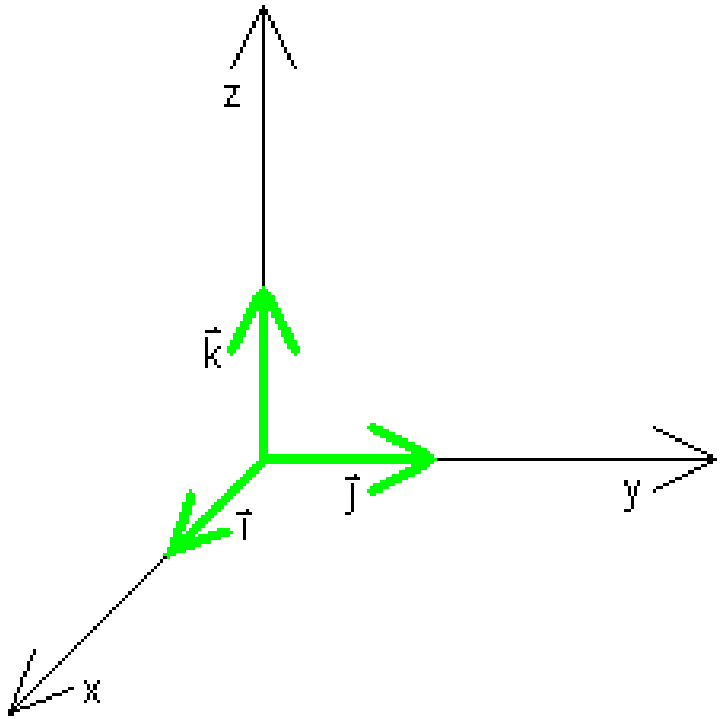
Neural Networks and Deep Learning



scikit-learn documentation

Vectors

A vector is a quantity with both a magnitude and a direction, usually represented by an arrow.



[HMC, 2014]

Vector Operations

Let $\vec{u} = (u_1, u_2, u_3)$ and $\vec{v} = (v_1, v_2, v_3)$.

Addition

$$\vec{u} + \vec{v} = (u_1 + v_1, u_2 + v_2, u_3 + v_3)$$

$$\vec{u} - \vec{v} = (u_1 - v_1, u_2 - v_2, u_3 - v_3)$$

Dot Product

Also called **scalar product** or **inner product**.

$$\vec{u} \cdot \vec{v} = u_1 v_1 + u_2 v_2 + u_3 v_3$$

Properties of the Dot Product:

- Commutative: $\vec{u} \cdot \vec{v} = \vec{v} \cdot \vec{u}$
- Distributive: $\vec{u} \cdot (\vec{v} + \vec{w}) = (\vec{u} \cdot \vec{v}) + (\vec{u} \cdot \vec{w})$

Vector **norm**, the magnitude of the vector, is: $||\vec{v}|| = \sqrt{\vec{v} \cdot \vec{v}}$

[HMC, 2014]

Vector Operations

Let $\vec{u} = (u_1, u_2, u_3)$ and $\vec{v} = (v_1, v_2, v_3)$.

Projection

The **projection** of $\vec{v}$ onto $\vec{u}$ is the portion of $\vec{v}$ that is parallel to $\vec{u}$.

The **dot product** gives the magnitude of the projection.

$$\vec{u} \cdot \vec{v} = ||\vec{u}|| \; ||\vec{v}|| cos(\theta)$$

$$\text{projection of } \vec{v} \text{ onto } \vec{u} : \frac{\vec{v} \cdot \vec{u}}{||u||^2} \vec{u}$$

$$\text{perpendicular vector component of } \vec{v} \text{ from } \vec{u} : \vec{v} - \frac{\vec{v} \cdot \vec{u}}{||u||^2} \vec{u}$$

Cross Product

The **cross product** $\vec{u} \times \vec{v}$ yields a vector perpendicular to both $\vec{u}$ and $\vec{v}$.

Properties of the Cross Product:

- $\vec{u} \times \vec{v} = -(\vec{v} \times \vec{u})$
- $\vec{u} \times (\vec{v} + \vec{w}) = (\vec{u} \times \vec{v}) + (\vec{u} \times \vec{w})$
- $\vec{u} \times \vec{u} = \vec{0}$

[HMC, 2014]

Matrices

A **matrix** is a two-dimensional rectangular array of numbers. An $n \times k$ matrix has n rows and k columns.

$$\begin{pmatrix} a_{11} & a_{12} & \dots & a_{1k} \\ a_{21} & a_{22} & \dots & a_{2k} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \dots & a_{nk} \end{pmatrix}$$

Terminology

- For an $n \times n$ **square** matrix A, the elements $a_{11}, a_{22}, \dots, a_{nn}$ for the **main diagonal** of the matrix.
- The **trace** of A is the sum of the main diagonal: $\sum_{i=1}^{n} a_{ii}$.
- The **transpose** of A is the matrix A^T formed by interchanging the rows and columns of A.
- If $A = A^T$, the matrix A is **symmetric**.

[HMC, 2014]

Matrix Operations

Let $A = [a_{ij}]$ and $B = [b_{ij}]$

Addition

If A and B are the same size, addition is defined. Each element of the new matrix is the element-wise sum of A and B.

$$A + B = [a_{ij} + b_{ij}]$$

Scalar Multiplication

$$cA = [ca_{ij}]$$

Matrix Multiplication

To multiply matrices A and B, the number of columns in A must be the same as the number of rows in B.

$$AB = [ab_{ij}] = [\sum_{k=1}^{n} a_{ik} b_{kj}]$$

[HMC, 2014]

Matrix Operations

Let $A = [a_{ij}]$ and $B = [b_{ij}]$

Matrix Inverse

The identity matrix I is the matrix with all main diagonal entries equal to 1, and all other elements equal to 0. This has the property:

$$AI = IA = A$$

The inverse of A is the matrix C such that:

$$AC = CA = I$$

and is denoted A^{-1}.

- Only square matrices can have inverses.
- Not every square matrix has an inverse. If an inverse exists, it is unique.
- If a matrix has an inverse, it is said to be **invertible**.

[HMC, 2014]

Change of Basis

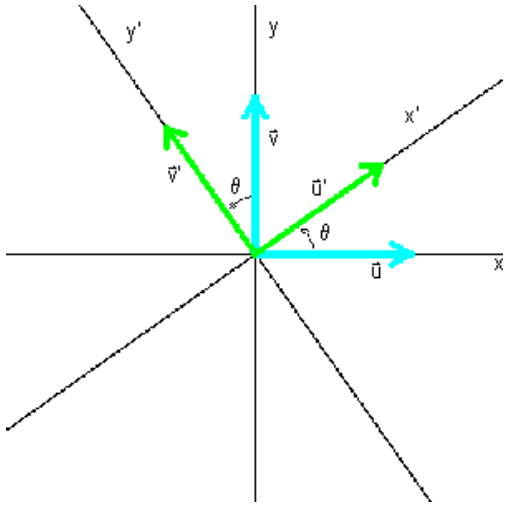
Matrices can form a map from one vector space to another.

Example: Rotation of Coordinates

The new basis $B' = \{\mathbf{u}', \mathbf{v}'\}$ of unit vectors along the $x'-$ and $y'-$ axies, respectively, has coordinates:

$$[\mathbf{u}']_\mathbf{B} = \begin{bmatrix} cos\theta \\ sin\theta \end{bmatrix}, [\mathbf{v}']_\mathbf{B} = \begin{bmatrix} -sin\theta \\ cos\theta \end{bmatrix}$$

in the original coordinate system B.



[HMC, 2014]

Change of Basis

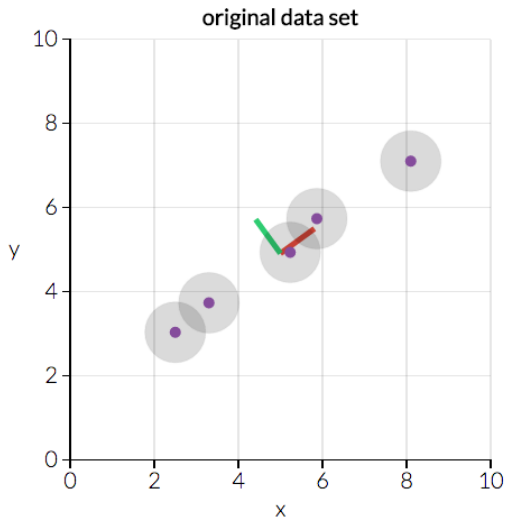
The matrix $P = \begin{bmatrix} cos\theta & -sin\theta \\ sin\theta & cos\theta \end{bmatrix}$ maps from the original coordinate system B to the rotated coordinate system B'.

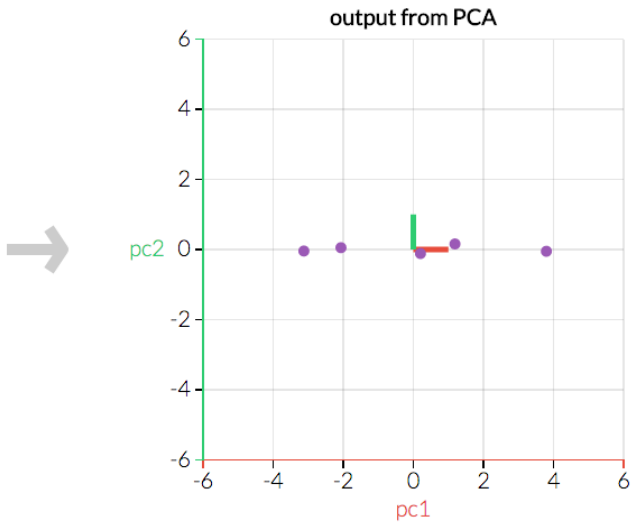
A vector $\begin{bmatrix} x \\ y \end{bmatrix}_B$ in the original coordinate system has coordinates $\begin{bmatrix} x' \\ y' \end{bmatrix}_{B'}$ in the new system.

$$\begin{bmatrix} x' \\ y' \end{bmatrix}_{B'} = \begin{bmatrix} cos\theta & -sin\theta \\ sin\theta & cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}_B$$

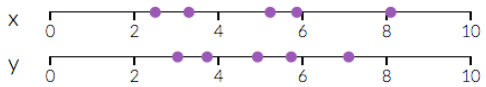
Principal Component Analysis

Principal Component Analysis (PCA) is a matrix-based method that finds the dimensions in a set of data that hold the most variability.





PCA is useful for eliminating dimensions. Below, we've plotted the data along a pair of lines: one composed of the x-values and another of the y-values.



If we're going to only see the data along one dimension, though, it might be better to make that dimension the principal component with most variation. We don't lose much by dropping PC2 since it contributes the least to the variation in the data set.

[Explained Visually, 2015]

Principal Component Analysis

Covariance matrix: $C_X = \frac{1}{n-1} XX^T$ where X is an $m \times n$ data matrix where each row is a set of n observations.

- C_X is a square, symmetric $m \times m$ matrix.
- The diagonal terms of C_X are the measure variance.
- The off-diagonal terms are the covariance between measurements.

Symmetric matrices are diagonalizable, and can be rewritten:

$$XX^T = S\Lambda S^{-1}$$

The principal component basis $Y = S^T X$ gives a new basis set, such that most of the variability in the data lies along the first direction.

References


Dale Hoffman (2016)
Contemporary Calculus, http://contemporarycalculus.com/


Joel Grus (2015)
Data Science from Scratch, O'Reilly


Andre Ng (2016)
Machine Learning Course - Stanford University Coursera


HMC Mathematics Online Tutorial (2014)
Harvey Mudd College


Principal Component Analysis (2015)
Explained Visually

Recommended Reading

Data Science From Scratch, Chapter 4 (linear algebra)
Data Science From Scratch, Chapter 8 (gradient descent)
Contemporary Calculus, Chapter 2

Articles for discussion:

Four Pitfalls of Hill Climbing

Examples

IPython notebook: Gradient Descent For Linear Regression
Gradient descent from scratc, by Matt Nedrichh via GitHub

IPython notebook: PCA (go over together)
Explained Visually website