Introduction to Data Science: Data Analysis with Geometry

Héctor Corrada Bravo

University of Maryland, College Park, USA

2020-04-05

Data Analysis with Geometry

A common situation:

- an outcome attribute (variable) Y, and
- one or more independent covariate or predictor attributes $X_1, \ldots, X_p$.

One usually observes these variables for multiple "instances" (or entities).

Data Analysis with Geometry

One may be interested in various things:

- What effects do the covariates X_i have on the outcome Y?
- How well can we quantify these effects?
- Can we predict outcome Y using covariates X_i?, etc...

Data Analysis with Geometry

Motivating Example: Credit Analysis

default	student	balance	income
No	No	729.5265	44361.625
No	Yes	817.1804	12106.135
No	No	1073.5492	31767.139
No	No	529.2506	35704.494
No	No	785.6559	38463.496
No	Yes	919.5885	7491.559

Data Analysis with Geometry

Task predict account default

What is the outcome Y?

What are the predictors X_j?

From data to feature vectors

The vast majority of ML algorithms we see in class treat instances as "feature vectors".

We can represent each instance as a *vector* in Euclidean space $\langle x_1, \ldots, x_p, y \rangle$.

From data to feature vectors

The vast majority of ML algorithms we see in class treat instances as "feature vectors".

We can represent each instance as a *vector* in Euclidean space $\langle x_1, \ldots, x_p, y \rangle$.

- every measurement is represented as a continuous value
- in particular, categorical variables become numeric (e.g., one-hot encoding)

From data to feature vectors

Here is the same credit data represented as a matrix of feature vectors

default	student	balance	income
1	0	1717.0716	38408.89
1	1	1983.2345	25687.93
-1	1	883.1573	18213.08
1	0	1975.6530	38221.84
-1	0	0.0000	32809.33
-1	0	528.0893	46389.34

Technical notation

- Observed values will be denoted in lower case. So x_i means the ith observation of the random variable X.

- Matrices are represented with bold face upper case. For example $\mathbf{X}$ will represent all observed predictors.

- N (or n) will usually mean the number of observations, or length of Y. i will be used to denote which observation and j to denote which covariate or predictor.

Technical notation

- Vectors will not be bold, for example x_i may mean all predictors for subject i, unless it is the vector of a particular predictor $\mathbf{x}_j$.
- All vectors are assumed to be column vectors, so the i-th row of $\mathbf{X}$ will be x_i', i.e., the transpose of x_i.

Geometry and Distances

Now that we think of instances as vectors we can do some interesting operations.

Let's try a first one: define a distance between two instances using Euclidean distance

$$d(x_1, x_2) = \sqrt{\sum_{j=1}^{p} (x_{1j} - x_{2j})^2}$$

Geometry and Distances

K-nearest neighbor classification

Now that we have a distance between instances we can create a classifier. Suppose we want to predict the class for an instance x.

K-nearest neighbors uses the closest points in predictor space predict Y.

$$\hat{Y} = \frac{1}{k} \sum_{x_k \in N_k(x)} y_k.$$

$N_k(x)$ represents the k-nearest points to x. How would you use $\hat{Y}$ to make a prediction?

Geometry and Distances

```
function KNN-CLASSIFY(x, X, y, K)
    S ← []                                    ▷ Compute distance to all points in X
    for all i = 1, ..., N do
        S ⊕ ⟨d(x, x_i), i⟩
    end for
    S ← sort(S)                               ▷ Find K nearest points
    ŷ ← 0
    for all k = 1, ..., K do
        ⟨d(x, x_i)⟩ ← S_k
        ŷ ← ŷ + y_i                           ▷ Update prediction
    end for
    return sign(ŷ)                            ▷ Return +1 if ŷ > 0, −1 otherwise
end function
```

Geometry and Distances

Inductive bias

The assumptions we make about our data that allow us to make predictions.

In KNN, our *inductive bias* is that points that are **nearby** will be of the same class.

Geometry and Distances

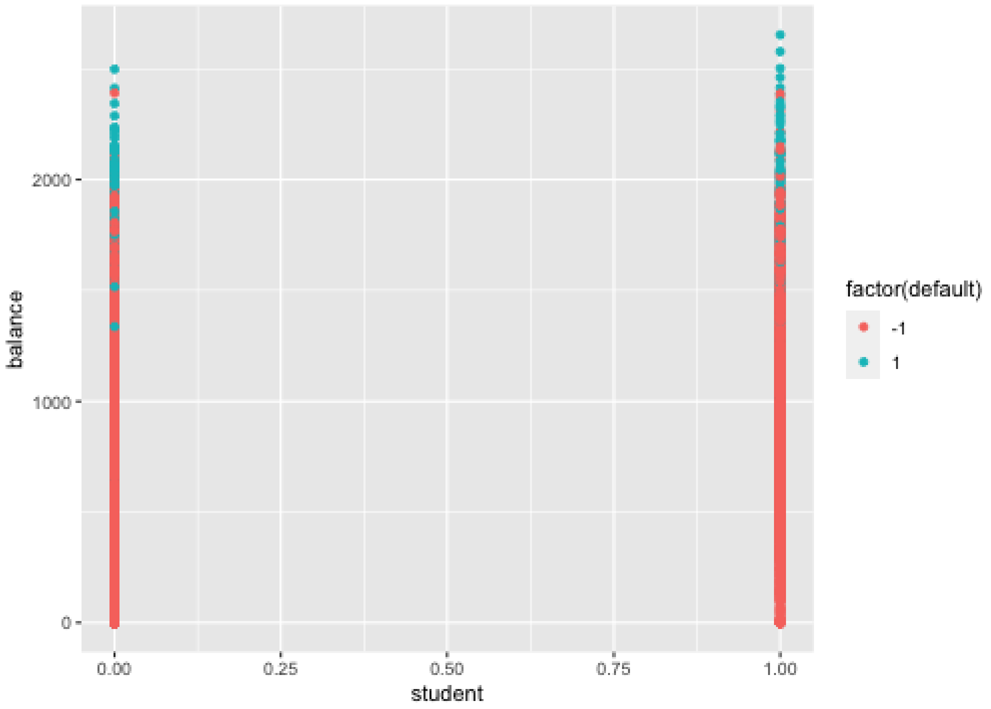
Parameter K is a *hyper-parameter*, it's value may affect prediction accuracy significantly.

Question: which situation may lead to *overfitting*, high or low values of K ? Why?

The importance of transformations

Feature scaling is an important issue in distance-based methods.

Which of these two features will affect distance the most?



Quick vector algebra review

- A (real-valued) vector is just an array of real values, for instance $x = \langle 1, 2.5, -6 \rangle$ is a three-dimensional vector.
- Vector sums are computed pointwise, and are only defined when dimensions match, so

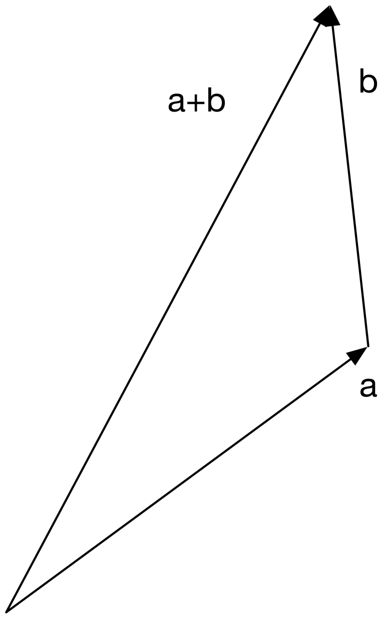
$$\langle 1, 2.5, -6 \rangle + \langle 2, -2.5, 3 \rangle = \langle 3, 0, -3 \rangle$$

.

In general, if $c = a + b$ then $cd = ad + bd$ for all vectors d.

Quick vector algebra review

Vector addition can be viewed geometrically as taking a vector a, then tacking on b to the end of it; the new end point is exactly c.



Quick vector algebra review

Scalar Multiplication: vectors can be scaled by real values;

$$2\langle 1, 2.5, -6\rangle = \langle 2, 5, -12\rangle$$

In general, $ax = \langle ax_1, ax_2, \ldots, ax_p\rangle$

Quick vector algebra review

The norm of a vector x, written $\|x\|$ is its length.

Unless otherwise specified, this is its Euclidean length, namely:

$$\|x\| = \sqrt{\sum_{j=1}^{p} x_j^2}$$

Quick vector algebra review

Quiz

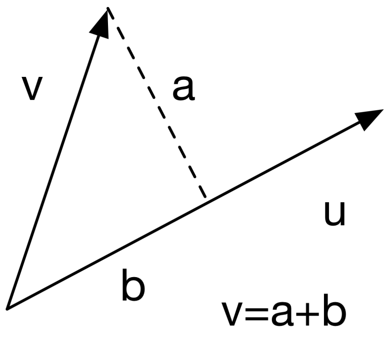
Write Euclidean distance of vectors u and v as a vector norm

Quick vector algebra review

The *dot product*, or *inner product* of two vectors u and v is defined as

$$u'v = \sum_{j=1}^{p} u_i v_i$$

A useful geometric interpretation of the inner product $v'u$ is that it gives the projection of v onto u (when $\|u\| = 1$).



The curse of dimensionality

Distance-based methods like KNN can be problematic in high-dimensional problems

Consider the case where we have many covariates. We want to use k-nearest neighbor methods.

Basically, we need to define distance and look for small multi-dimensional "balls" around the target points.

With many covariates this becomes difficult.

Summary

- We will represent many ML algorithms geometrically as vectors
- Vector math review
- K-nearest neighbors
- The curse of dimensionality