CHAPTER 2

Mathematical Review

Prior to discussing machine learning, a brief overview of statistics is necessary. Broadly, *statistics* is the analysis and collection of quantitative data with the ultimate goal of making actionable insights on this data. With that being said, although machine learning and statistics aren't the same field, they are closely related. This chapter gives a brief overview of terms relevant to our discussions later in the book.

Statistical Concepts

No discussion about statistics or machine learning would be appropriate without initially discussing the concept of probability.

Probability

Probability is the measure of the likelihood of an event. Although many machine learning models tend to be deterministic (based off of algorithmic rules) rather than probabilistic, the concept of probability is referenced specifically in algorithms such as the expectation maximization algorithm in addition to more complex deep learning architectures such a recurrent neural networks and convolutional neural networks. Mathematically, this algorithm is defined as the following:

$$Probability\ of\ Event\ A = \frac{number\ of\ times\ event\ A\ occurs}{all\ possible\ events}$$

This method of calculating probability represents the *frequentist* view of probability, in which probability is by and large derived from the following formula. However, the other school of probability, Bayesian, takes a differing approach. Bayesian probability theory is based on the assumption that probability is conditional. In other words, the likelihood of an event is influenced by the conditions that currently exist or events that have happened prior. We define conditional probability in the following equation. The probability of an event A, given that an event B has occurred, is equal to the following:

$$P(A|B) = \frac{P(A \cap B)}{P(B)},$$

$$\textit{Provided } P(B) > 0.$$

In this equation, we read $P(A|B)$ as "the probability of A given B" and $P(A \cap B)$ as "the probability of A and B."

With this being said, calculating probability is not as simple as it might seem, in that dependency versus independency must often be evaluated. As a simple example, let's say we are evaluating the probability of two events, A and B. Let's also assume that the probability of event B occurring is dependent on A occurring. Therefore, the probability of B occurring should A not occur is 0. Mathematically, we define dependency versus independency of two events A and B as the following:

$$P(A|B) = P(A)$$

$$P(B|A) = P(B)$$

$$P(A \cap B) = P(A)P(B)$$

In Figure 2-1, we can envision events A and B as two sets, with the union of A and B as the intersection of the circles:

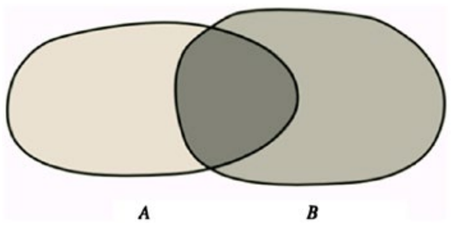

Figure 2-1. *Representation of two events (A,B)*

Should this equation not hold in a given circumstance, the events A and B are said to be dependent.

And vs. Or

Typically when speaking about probability—for instance, when evaluating two events A and B—probability is often in discussed in the context of "the probability of A *and* B" or "the probability of A *or* B." Intuitively, we define these probabilities as being two different events and therefore their mathematical derivations are difference. Simply stated, *or* denotes the addition of probabilities events, whereas *and* implies the multiplication of probabilities of event. The following are the equations needed:

And (multiplicative law of probability) is the probability of the intersection of two events A and B:

$$P(A \cap B) = P(A)P(B|A)$$

$$= P(B)P(A|B)$$

If the events are independent, then

$$P(A \cap B) = P(A)P(B)$$

Or (additive law of probability) is the probability of the union of two events A and B:

$$P(A \cup B) = P(A) + P(B) - P(A \cap B)$$

The symbol $P(A \cup B)$ means “the probability of A or B.”

Figure 2-2 illustrates this.

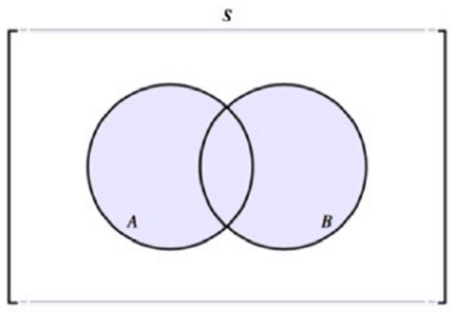

Figure 2-2. *Representation of events A,B and set S*

The probabilities of A *and* B exclusively are the section of their respective spheres which do not intersect, whereas the probability of A *or* B would be the addition of these two sections plus the intersection. We define S as the sum of all sets that we would consider in a given problem plus the space outside of these sets. The probability of S is therefore always 1.

With this being said, the space outside of A and B represents the opposite of these events. For example, say that A and B represent the probabilities of a mother coming home at 5 p.m. and a father coming home at 5 p.m. respectively. The white space represents the probability that neither of them comes home at 5 p.m.

Bayes' Theorem

As mentioned, Bayesian statistics is continually gaining appreciation within the fields of machine learning and deep learning. Although these techniques can often require considerable amounts of hard coding, their power comes from the relatively simple theoretical underpinning while being powerful and applicable in a variety of contexts. Built upon the concept of conditional probability, Bayes' theorem is the concept that the probability of an event A is related to the probability of other similar events:

$$P\left(B_j \mid A\right) = \frac{P\left(A \mid B_j\right)P\left(B_j\right)}{\sum_i^k P\left(A \mid B_i\right)P\left(B_i\right)}$$

Referenced in later chapters, Bayesian classifiers are built upon this formula as well as the expectation maximization algorithm.

Random Variables

Typically, when analyzing the probabilities of events, we do so within a set of random variables. We define a random variable as a quantity whose value depends on a set of possible random events, each with an associated probability. Its value is known prior to it being drawn, but it also can be defined as a function that maps from a probability space. Typically, we draw these random variables via a method know as random sampling. *Random sampling* from a population is said to be random when each observation is chosen in such a way that it is just as likely to be selected as the other observations within the population.

Broadly speaking, the reader can expect to encounter two types of random variables: *discrete random variables* and *continuous random variables*. The former refers to variables that can only assume a finite number of distinct values, whereas the latter are variables that have an infinite number of possible variables. An example is the number of cars in a garage versus the theoretical change in percentage change of a stock price. When analyzing these random variables, we typically rely on a variety of statistics that readers can expect to see frequently. But these statistics often are used directly in the algorithms either during the various steps or in the process of evaluating a given machine learning or deep learning model.

As an example, arithmetic means are directly used in algorithms such as K-means clustering while also being a theoretical underpinning of the model evaluation statistics such as mean squared error (referenced later in this chapter). Intuitively, we define the arithmetic mean as the central tendency of a discrete set of numbers—specifically it is the sum of the values divided by the number of the values. Mathematically, this equation is given by the following:

$$\bar{x} = \frac{1}{N}\sum_{i=1}^{N} x_i$$

The *arithmetic* mean, broadly speaking, represents the most likely value from a set of values within a random variable. However, this isn't the only type of mean we can use to understand a random variable. The *geometric* mean is also a statistic that describes the central tendency of a sequence of numbers, but it is acquired by using the product of the values rather than the sum. This is typically used when comparing different items within a sequence, particularly if they have multiple properties individually. The equation for the geometric mean is given as follows:

$$\left(\prod_{i=1}^{n} x_i\right)^{\frac{1}{n}} = \left(x_1 * x_2 * \ldots * x_n\right)^{\frac{1}{n}}$$

For those involved in fields where the use of time series is frequent, geometric means are useful to acquiring a measure of change over certain intervals (hours, months, years, and so on). That said, the central tendency of a random variable is not the only useful statistic for describing data. Often, we would like to analyze the degree to which the data is dispersed around the most probable value. Logically, this leads us to the discussion of variance and standard deviation. Both of these statistics are highly related, but they have a few key distinctions: *variance* is the squared value of standard deviation, and the standard deviation is often more referenced than variance across various fields. When addressing the latter distinction, this is because variance is much harder to visually describe, in addition to the fact that the units that variance is in are ambiguous. Standard deviation is in the units of the random variable being analyzed and is easy to visualize.

For example, when evaluating the efficiency of a given machine learning algorithm, we could draw the mean squared error from several epochs. It might be helpful to collect sample statistics of these variables, such that we can understand the dispersion of this statistic. Mathematically, we define variance and standard deviation as the following

Variance

$$\sigma^2 = \frac{\Sigma\left(X-\mu\right)^2}{N}$$

$$Var(X) = E\left[\left(X - E([X])\right)^2\right]$$

$$= E[X^2] - 2XE[X] + \left(E[X]\right)^2$$

$$= E\left[X^2\right] - 2E[X]E[X] + \left(E[X]\right)^2$$

$$= E\left[X^2\right] - 2E[X]E[X] + \left(E[X]\right)^2$$

Standard Deviation

$$\sigma = \sqrt{\left(\frac{\sum_{i}^{n} (x_i - \bar{x})^2}{n-1} \right)}$$

Also, covariance is useful for measuring the degree to which a change in one feature affects the other. Mathematically, we define covariance as the following:

$$cov(X,Y) = \frac{1}{n} \sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y})$$

Although deep learning has made significant progress in modeling relationships between variables with non-linear correlations, some estimators that one would use for more simple tasks require this as a preliminary assumption. For example, linear regression requires this to be an assumption, and although many machine learning algorithms can model complex data, some are better at it than others. As such, it is recommended that prior to selecting estimators features be examined for their relationship to one another using these prior statistics. As such, this leads us to the discussion of the correlation coefficient which measures the degree to which to variables are linearly related to each other. Mathematically, we define this as follows:

$$correlation = \rho = \frac{1}{n} \sum_{i=1}^{n} \frac{(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{(x_i - \bar{x})^2 (y_i - \bar{y})^2}}$$

Correlation coefficients can have a value as low as –1 and as high as 1, with the lower bound representing an *opposite* correlation and the upper bound representing *complete* correlation. A correlation coefficient of 0 represents complete lack of correlation, statistically speaking. When evaluating machine learning models, specifically those that perform regression, we typically reference the coefficient of determination (R squared) and mean squared error (MSE). We think of *R squared* as a measure of how well the estimated regression line of the model fits the distribution of the data. As such, we can state that this statistic is best known as the *degree of fitness* of a given model. MSE measures the average of the squared error of the deviations from the models predictions to the observed data. We define both respectively as the following:

Coefficient of Determination (R Squared)

$$R^2 = 1 - \sum_{i}^{n} \frac{\left(\hat{y}_i - y\right)^2}{\left(\hat{y}_i - \overline{y}\right)^2}$$

Mean Squared Error (MSE)

$$MSE = \frac{1}{n} \sum_{i=1}^{n} \left(y_i - \overline{y}\right)^2$$

With respect to what these values should be, I discuss that in detail later in the text. Briefly stated, though, we typically seek to have models that have high R squared values and lower MSE values than other estimators chosen.

Linear Algebra

Concepts of linear algebra are utilized heavily in machine learning, data science, and computer science. Though this is not intended to be an exhaustive review, it is appropriate for all readers to be familiar with the following concepts at a minimum.

Scalars and Vectors

A *scalar* is a value that only has one attribute: *magnitude*. A collection of scalars, known as a vector, can have both magnitude and direction. If we have more than one scalar in a given vector, we call this an *element of vector space*. Vector space is distinguished by the fact that it is sequence of scalars that can be added and multiplied, and that can have other numerical operations performed on them. *Vectors* are defined as a column vector of n numbers. When we refer to the indexing of a vector, we will describe *i* as the index value. For example, if we have a vector x, then x_1 refers to the first value in vector x. Intuitively, imagine a vector as an object similar to a file within a file cabinet. The values within this vector are the individual sheets of paper, and the vector itself is the folder that holds all these values.

Vectors are one of the primary building blocks of many of the concepts discussed in this text (see Figure 2-3). For example, in deep learning models such as Doc2Vec and Word2Vec, we typically represent words, and documents of text, as vectors. This representation allows us to condense massive amount of data into a format easy to input to neural networks to perform calculations on. From this massive reduction of dimensionality, we can determine the degree of similarity, or dissimilarity, from one document to another, or we can gain better understanding of synonyms than from simple Bayesian inference. For data that is already numeric, vectors provide an easy method of "storing" this data to be inputted into algorithms for the same purpose. The properties of vectors (and matrices), particularly with respect to mathematical operations, allow for relatively quick calculations to be performed over massive amounts of data, also presenting a computational advantage of manually operating on each individual value within a data set.

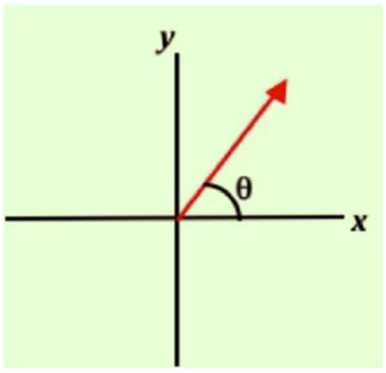

Figure 2-3. *Representation of a vector*

Properties of Vectors

Vector dimensions are often denoted by $\mathbb{R}^n$ or $\mathbb{R}^m$ where n and m is the number of values within a given vector. For example, $x \in \mathbb{R}^5$ denotes set of 5 vectors with real components. Although I have only discussed a column vector so far, we can also have a row vector. A transformation to change a column vector into a row vector can also be performed, known as a transposition. A *transposition* is a transformation of a matrix/vector X such that the rows of X are written as the columns of X^T and the columns of X are written as the rows of X^T.

Addition

Let's define two vectors $d = [d_1, d_2, \ldots, d_n]^T$ and $e = [e_1, e_2, \ldots, e_n]^T$ where

$$d_n = e_n, \, for \; i = 1, 2, \ldots, n$$

The sum of the vectors is therefore the following:

$$d + e = [(d_1 + e_1), (e_2 + d_2), \ldots, (d_n + e_n)]^T$$

Subtraction

Given that the assumptions from the previous example have not changed, the difference between vectors d and e would be the following:

$$d - e = [(d_1 - e_1), (e_2 - d_2), \ldots, (d_n - e_n)]^T$$

Element Wise Multiplication

Given that the assumptions from the previous example have not changed, the product of vectors d and e would be the following:

$$d * e = \left[(d_1 * e_1), (e_2 * d_2), \ldots, (d_n * e_n) \right]^T$$

Axioms

Let *a*,*b*, and *x* be a set of vectors within set *A*, and *e* and *d* be scalars in B. The following axioms must hold if something is to be a vector space:

Associative Property

The associative property refers to the fact that rearranging the parentheses in a given expression will not change the final value:

$$x + (a + b) = (x + a) + b$$

Commutative Property

The commutative property refers to the fact that changing the order of the operands in a given expression will not change the final value:

$$a + b = b + a$$

Identity Element of Addition

$$a + 0 = a, \textit{ for all } a \in A$$

Where $0 \in A$. 0 in this instance is the zero vector, or a vectors of zeros.

Inverse Elements of Addition

In this instance, for every a := A, there exists an element –a := A, which we label as the additive inverse of a:

$$a + (-a) = 0$$

Identity Element of Scalar Multiplication

$$(1)a = a$$

Distributivity of Scalar Multiplication with Respect to Vector Addition

$$e(a+b) = ea + eb$$

Distributivity of Scalar Multiplication with Respect to Field Addition

$$(a+b)d = ad + bd$$

Subspaces

A *subspace* of a vector space is a nonempty subset that satisfies the requirements for a vector space, specifically that linear combinations stay in the subspace. This subset is "closed" under addition and scalar multiplication. Most notably, the zero vector will belong to every subspace. For example, the space that lies between the hyperplanes of produced by a support vector regression, a machine learning algorithm I address later, is an example of a subspace. In this subspace are acceptable values for the response variable.

Matrices

A matrix is another fundamental concept of linear algebra in our mathematical review. Simply put, a *matrix* is a rectangular array of numbers, symbols, or expressions arranged in rows and columns. Matrices have a variety of uses, but specifically are often used to store numerical data. For example, when performing image recognition with a convolutional neural network, we represent the pixels in the photos as numbers within a 3-dimensional matrix, representing the matrix for the red, green, and blue photos comprised of a color photo. Typically, we take an individual pixel to have 256 individual values, and from this mathematical interpretation an otherwise difficult-to-understand representation of data becomes possible. In relation to vectors and scalars, a matrix contains scalars for each individual value and is made up of row and column vectors. When we are indexing a given matrix A, we will be using the notation A_{ij}. We also say that $A = a_{ij},\ A \in \mathbb{R}^{m\,x\,n}$.

Matrix Properties

Matrices themselves share many of the same elementary properties that vectors have by definition of matrices being combinations of vectors. However, there are some key differences that are important, particularly with respect to matrix multiplication. For example, matrix multiplication is a key element of understanding how ordinary least squares regression works, and fundamentally why we would be interested in using gradient descent when performing linear regression. With that being said, the properties of matrices are discussed in the rest of this section.

Addition

Let's assume A and B are both matrices with *m* x *n* dimensions:

$$A + B = \left(A_{ij} + B_{ij}\right), \, for \; i = 1,2,\ldots,n$$

Scalar Multiplication

Let us assume A and B are both matrices with *m* x *n* dimensions

$$AB = \left(A_{ij} * B_{ij}\right), \, for \; i = 1,2,\ldots,n$$

Transposition

$$A_{ij}^{T} = A_{ji}$$

Types of Matrices

Matrices come in multiple forms, usually denoted by the shape that they take on. Although a matrix can take on a multitude of dimensions, there are many that will commonly references. Among the simplest is the square matrix, which is distinguished by the fact that it has an equal amount of rows and columns:

$$\mathbf{A} = \begin{bmatrix} a_{1,1} & a_{1,2} \; a_{1,3} \cdots & a_{1,n} \\ \vdots & \ddots & \vdots \\ a_{n,1} & a_{n,2} a_{n,3} \cdots & a_{n,n} \end{bmatrix}$$

It is generally unlikely that the reader will come across a square matrix, but the implications of matrix properties make discussing it necessary. That said, this brings us to discussing different types of matrices such as the diagonal and identity matrix. The *diagonal matrix* is a matrix where all the entries that are not along the main diagonal of the matrix (from the top left corner through the bottom right corner) are zero, given by the following:

$$A = \begin{matrix} 5 & 0 & 0 \\ 0 & 4 & 0 \\ 0 & 0 & 3 \end{matrix}$$

Similar to the diagonal matrix, the *identity matrix* also has zeros for values along all entries except for the diagonal of the matrix. The key distinction here, however, is that all the entries in the diagonal matrix are 1. This matrix is given by the following diagram:

$$I_n = \begin{matrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{matrix}$$

Another matrix you're not likely to see, but which is important from a theoretical perspective, is the symmetric matrix, whose transpose is equal to the non-transformed matrix. I describe *transpose* subsequently in this chapter, but it can be understood simply as transforming the rows into the columns and vice versa.

The final types of matrix I will define, specifically referenced in Newton's method (an optimization method described in Chapter 3), are definite and semi-definite matrices. A symmetric matrix is called positive-definite if all entries are greater than zero. But if all the values are all non-negative, the matrix is called *positive semi-definite.* Although described in greater detail in the following chapter, this is important for the purpose of understanding whether a problem has a global optimum (and therefore whether Newton's method can be used to find this global optimum).

Matrix Multiplication

Unlike vectors, matrix multiplication contains unique rules that will be helpful for readers who plan on applying this knowledge, particularly those using programming languages. For example, imagine that we have two matrices, A and B, and that we want to multiply them. These matrices can only be multiplied under the condition that the number of columns in A is the same as the number of rows in column B. We call this matrix product the *dot product* of matrices A and B. The next sections discuss examples of matrix multiplication and its products.

Scalar Multiplication

Assume we have some matrix, A, that we would like to multiply by the scalar value sigma. The result of this operation is displayed by the following diagram:

$$\sigma A = \sigma \begin{bmatrix} A_{1,1} & A_{1,2} & \cdots & A_{1,m} \\ \vdots & & \ddots & \vdots \\ A_{n,1} & A_{n,2} & \cdots & A_{n,m} \end{bmatrix} = \begin{bmatrix} \sigma A_{1,1} & \sigma A_{1,2} & \cdots & \sigma A_{1,m} \\ \vdots & & \ddots & \vdots \\ \sigma A_{n,1} & \sigma A_{n,2} & \cdots & \sigma A_{n,m} \end{bmatrix}$$

Each value in the matrix is multiplied by the scalar such in the new matrix that is subsequently yielded. Specifically, we can see this relationship displayed in the equations following related to eigendecomposition.

Matrix by Matrix Multiplication

Matrix multiplication is utilized in several regression methods, specifically OLS, ridge regression, and LASSO. It is an efficient yet simple way of representing mathematical operations on separate data sets. In the following example, let D be an *n* x *m* matrix and E be an *m* x *p* matrix such that when we multiply them both by each other, we get the following:

$$D = \begin{bmatrix} D_{1,1} & D_{1,2} & \cdots & D_{1,m} \\ \vdots & & \ddots & \vdots \\ D_{n,1} & D_{n,2} & \cdots & D_{n,m} \end{bmatrix}, E = \begin{bmatrix} E_{1,1} & E_{1,2} & \cdots & E_{1,p} \\ \vdots & & \ddots & \vdots \\ E_{m,1} & E_{n,2} & \cdots & E_{m,p} \end{bmatrix}$$

$$DE = \begin{bmatrix} DE_{1,1} & DE_{1,2} & \cdots & DE_{1,p} \\ \vdots & & \ddots & \vdots \\ DE_{n,1} & DE_{n,2} & \cdots & DE_{n,p} \end{bmatrix}$$

Assuming that the dimensions are equal, each element in one matrix is multiplied by the corresponding element of the other element, yielding a new matrix. Although walking through these examples may seem pointless, it is actually more important than it appears—particularly because all the operations will be performed by a computer. Readers should be familiar with, if only for the purpose of debugging errors in code, the products of matrix multiplication. We will see different matrix operations that also will occur in different contexts later.

Row and Column Vector Multiplication

For those wondering how exactly matrix multiplication yields a single scalar value, the following section elaborates on this further. If

$$\boldsymbol{X} = \begin{pmatrix} \boldsymbol{x} & \boldsymbol{y} & \boldsymbol{z} \end{pmatrix}, \quad \boldsymbol{Y} = \begin{matrix} \boldsymbol{d} \\ \boldsymbol{e} \\ \boldsymbol{f} \end{matrix}$$

then their matrix products are given by the following:

$$\boldsymbol{XY} = \begin{pmatrix} \boldsymbol{x} & \boldsymbol{y} & \boldsymbol{z} \end{pmatrix} \begin{matrix} \boldsymbol{d} \\ \boldsymbol{e} \\ \boldsymbol{f} \end{matrix}$$

$$\boldsymbol{XY} = \boldsymbol{xd} + \boldsymbol{ye} + \boldsymbol{zf}$$

Contrastingly:

$$\boldsymbol{YX} = \begin{matrix} \boldsymbol{d} \\ \boldsymbol{e} \\ \boldsymbol{f} \end{matrix} \begin{pmatrix} \boldsymbol{x} & \boldsymbol{y} & \boldsymbol{z} \end{pmatrix}$$

$$\boldsymbol{YX} = \begin{matrix} \boldsymbol{dx} & \boldsymbol{dy} & \boldsymbol{dz} \\ \boldsymbol{ex} & \boldsymbol{ey} & \boldsymbol{ez} \\ \boldsymbol{fx} & \boldsymbol{fy} & \boldsymbol{fz} \end{matrix}$$

Column Vector and Square Matrix

In some cases, we need to multiple a column vector by an entire matrix. In this instance, the following holds:

$$\boldsymbol{B} = \begin{matrix} \mathbf{1} & \mathbf{2} & \mathbf{3} \\ \mathbf{4} & \mathbf{5} & \mathbf{6} \\ \mathbf{7} & \mathbf{8} & \mathbf{9} \end{matrix}, \quad \boldsymbol{C} = \begin{matrix} \boldsymbol{d} \\ \boldsymbol{e} \\ \boldsymbol{f} \end{matrix}$$

The matrix product of B and C is given by the following:

$$\boldsymbol{YX} = \begin{matrix} \mathbf{1}\boldsymbol{d} & \mathbf{2}\boldsymbol{d} & \mathbf{3}\boldsymbol{d} \\ \mathbf{4}\boldsymbol{e} & \mathbf{5}\boldsymbol{e} & \mathbf{6}\boldsymbol{e} \\ \mathbf{7}\boldsymbol{f} & \mathbf{8}\boldsymbol{f} & \mathbf{9}\boldsymbol{f} \end{matrix}$$

Square Matrices

Among the simplest of matrix operations is when we are dealing with two square matrices, as follows:

$$\boldsymbol{B} = \begin{matrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{matrix}, \quad \boldsymbol{D} = \begin{matrix} 9 & 8 & 7 \\ 6 & 5 & 4 \\ 3 & 2 & 1 \end{matrix}$$

$$\boldsymbol{BD} = \begin{matrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{matrix} \; x \; \begin{matrix} 9 & 8 & 7 \\ 6 & 5 & 4 \\ 3 & 2 & 1 \end{matrix}$$

$$= \begin{matrix} (1*9)+(2*6)+(3*3) & (1*8)+(2*5)+(3*2) & (1*7)+(2*4)+(3*1) \\ (4*9)+(5*6)+(6*3) & (4*8)+(5*5)+(6*2) & (4*7)+(5*4)+(6*1) \\ (7*9)+(8*6)+(9*3) & (7*8)+(8*5)+(9*2) & (7*7)+(8*4)+(9*1) \end{matrix}$$

$$\boldsymbol{BD} = \begin{matrix} 30 & 24 & 18 \\ 84 & 69 & 54 \\ 138 & 114 & 90 \end{matrix}$$

By this same logic:

$$\boldsymbol{DB} = \begin{matrix} 90 & 114 & 138 \\ 54 & 69 & 84 \\ 18 & 24 & 30 \end{matrix}$$

Row Vector, Square Matrix, and Column Vector

In other cases, we will perform operations on matrices/vectors with distinct shapes among each:

$$\mathbf{A} = \begin{matrix} 9 & 8 & 7 \\ 6 & 5 & 4 \\ 3 & 2 & 1 \end{matrix}, \quad \mathbf{B} = \begin{matrix} 1 & 2 & 3 \end{matrix}, \quad \mathbf{C} = \begin{matrix} 4 \\ 5 \\ 6 \end{matrix}$$

$$\mathbf{ABC} = \begin{matrix} 9 & 8 & 7 \\ 6 & 5 & 4 \\ 3 & 2 & 1 \end{matrix} \; x \; \begin{matrix} 1 & 2 & 3 \end{matrix} \; x \; \begin{matrix} 4 \\ 5 \\ 6 \end{matrix}$$

$$\begin{matrix} 9 & 8 & 7 \\ 6 & 5 & 4 \\ 3 & 2 & 1 \end{matrix} \;\mathbf{X}\; \begin{matrix} 4 & 10 & 18 \end{matrix}$$

$$ABC = \begin{matrix} 36 & 32 & 28 \\ 60 & 50 & 40 \\ 54 & 36 & 18 \end{matrix}$$

Rectangular Matrices

Our last examples address the rectangular matrix. For this example, we have two matrices Z and Y such that:

$$\mathbf{Z} = \begin{matrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{matrix}, \quad \mathbf{Y} = \begin{matrix} 9 & 8 \\ 7 & 6 \\ 5 & 4 \end{matrix}$$

$$\mathbf{ZY} = \begin{matrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{matrix} \; x \; \begin{matrix} 9 & 8 \\ 7 & 6 \\ 5 & 4 \end{matrix}$$

$$= \begin{matrix} 9 & 40 \\ 28 & 18 \\ 10 & 240 \end{matrix}$$

Matrix Multiplication Properties (Two Matrices) Not Commutative

In general, given two matrices A and B, AB ≠ BA, AB and BA may not be simultaneously defined, and even if they are, they still may not be equal. This is contrary to ordinary multiplication of numbers. For example, to specify the ordering of matrix multiplication verbally, pre-multiply A by B means BA while post-multiply A by C means AC. As long as the entries of the matrix come from a ring that has an identity and $n > 1$, there is a pair of n x n non-commuting matrices over the ring. A notable exception is the identity matrix, because it commutes with every square matrix.

Distributive over Matrix Addition

Distributivity in matrices follows the same logic as it does in vectors. As such, the following axioms hold:

Left distributivity:

$$A(B+C)=AB+BC$$

Right distributivity:

$$(A+B)C=AC+BC$$

Index notation of these operations respectively are the following:

$$\sum_k A_{ik}\left(B_{kj}+C_{kj}\right)=\sum_k A_{ik}B_{kj}+\sum_k A_{ik}C_{kj}$$

$$\sum_k \left(A_{ik}+B_{ik}\right)C_{kj}=\sum_k A_{ik}C_{kj}+\sum_k B_{ik}C_{kj}$$

Scalar Multiplication Is Compatible with Matrix Multiplication

Following our discussion earlier of scalar multiplication with respect to a matrix, we see here that distributivity of scalar multiplication with matrices also holds. For example, we have the following equation, which proves this as such:

$$\lambda(AB)=(\lambda A)B$$

$$(AB)\lambda=A(B\lambda)$$

λ is a scalar. If the entries of the matrix are real or complex numbers, then all four quantities are equal. More generally, all four are equal if lambda belongs to the center of the ring of entries of the matrix, because in this case $\lambda X=X\lambda$.

Index notation of this is the following:

$$\lambda \sum_K \left(A_{ik}B_{kj}\right)=\sum_k \left(\lambda A_{ik}\right)B_{kj}=\sum A_{ik}\left(\lambda B_{kj}\right)$$

$$\sum_k \left(A_{ik}B_{kj}\right)\lambda=\sum \left(A_{ik}\lambda\right)B_{kj}=\sum_k A_{ik}\left(B_{kj}\lambda\right)$$

Transpose

As referred to earlier, the *transpose* of a matrix is an operation on a matrix where the product of this transformation is a new matrix in which the new matrix's rows are the original matrix's columns and the new matrix's columns are the original matrix's rows. The following equation shows how we denote this transformation, given two matrices A and B

$$(AB)^T = B^T A^T$$

where T denotes the transpose, the interchange of row I with column I in a matrix. This identity holds for any matrices over a commutative ring, but not for all rings in general. Note that A and B are reversed.

Index notation:

$$\begin{aligned}\left[(AB)^T\right]_{ij} &= (AB)_{ji} \\ &= \Sigma_K (A)_{jk} (B)_{ki} \\ &= \Sigma_k \left(A^T\right)_{kj} \left(B^T\right)_{ik} \\ &= \Sigma_k \left(B^T\right)_{ik} \left(A^T\right)_{kj} \\ &= \left[\left(B^T\right)\left(A^T\right)\right]_{ij}\end{aligned}$$

Trace

The *trace* of a product AB is independent of the order of A and B. The trace can also be thought of as the diagonal of a matrix:

$$tr(AB) = tr(BA)$$

Index notation:

$$\begin{aligned}tr(AB) &= \Sigma_i \Sigma_k A_{ik} B_{ik} \\ &= \Sigma_k \Sigma_i B_{ki} A_{ik} \\ &= tr(BA)\end{aligned}$$

Norms

A *norm* is a function that assigns a strictly positive length or size to each vector in a vector space. In machine learning you will encounter many various norms, and they play a vital role in reducing the MSE of regression models in addition to increasing accuracy in classification models. For example, ridge regression uses an L2 norm to shrink the regression coefficients during periods of high multicollinearity, and LASSO uses an L1 norm to shrink some regression coefficients to zero. I will review both of these regression models in detail in Chapter 3.

In the context of deep learning, experimentation of adding different layers in deep neural networks, in which norms are used to perform dimensionality reduction on data, have proved successful at some tasks. For example, use of an L2 norm layer was performed in a convolutional neural network. But this also can be used as a dissimilarity/loss measure in multilayer perceptrons rather than a traditional gradient function.

Euclidean Norm

This describes the distance over a vector within Euclidean space in $\mathbb{R}^n$. Let's assume $x = (x_1, x_2, \ldots, x_n)$.

L2 Norm

This gives the distance from the origin point within the vector to the last point within x, and is often referred to as the L2 norm:

$$\|x\|_2^2 = \sqrt{x_1^2 + x_2^2 + \ldots + x_n^2}$$

L1 Norm

This is the same equation as the L2 norm except that the scalars are not squared:

$$\|x\| = \sqrt{x_1 + x_2 + \ldots + x_n}$$

The shape of the L1 verses L2 norms are distinguished as shown in Figure 2-4.

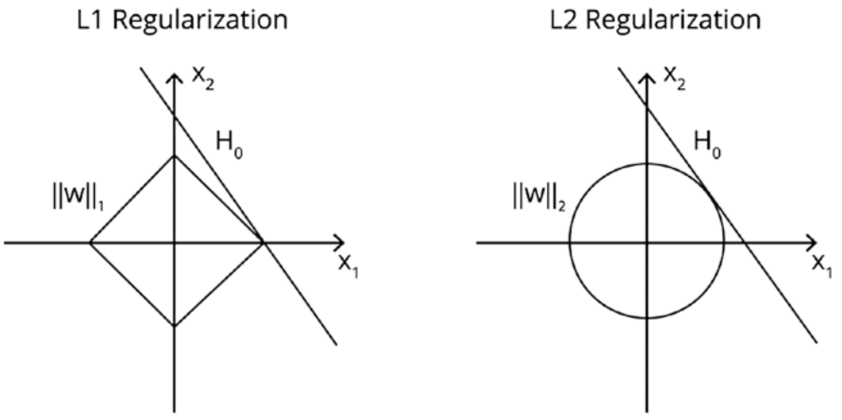

Figure 2-4. *L1 and L2 norm shapes*

Note that with the L1 norm we observe a square (or cubic) shape, and with the L2 norm we observe a circle (or spherical) shape. In certain situations, it's optimal to use the L1 norm to perform variable selection at the same time while also performing regression analysis, but this is an issue that isn't always present, and I discuss it in further detail in Chapter 8.

The advantage of using an L1 norm is obvious in that you can perform feature selection while performing regression. However, it should be noted that performing feature selection after reduction of the data set has already occurred can encourage overfitting. Strategies and general practices for building a robust model are reviewed more extensively in Chapter 8, but it is generally suggested that readers use the L1 norm in the instance that there has been little to no feature selection performed prior to fitting the data to the model.

For those interested in vehicle routing problems, the *taxicab (Manhattan) norm* is relevant for those who want to focus on fields related to transportation and/or delivery or packages/persons. The taxicab norm describes the distance a taxicab would travel along a given city block:

$$\|x\| = \Sigma_i |x_i|, \, for \; i = 1,2,\ldots,n$$

The absolute value norm is a norm on the one-dimensional vector spaces formed by real or complex numbers. Absolute value norms have been used in place of other loss functions or dissimilarity functions:

$$\|x\| = |x|$$

P-norm

Let p ≥ 1 be a real number:

$$\|x\|_p = \Sigma\left(|x_i|^p\right)^{\frac{1}{p}}$$

The shape of this norm is shown in Figure 2-5.

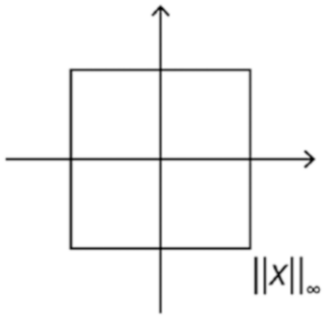

Figure 2-5. *P-norm*

For $p = 1,$ we get the taxicab norm, for $p = 2,$ we get the Euclidean norm, and as $p \to \infty,$ we get the infinity norm or the maximum norm. The p-norm is related to the generalized mean or power mean. When $0 < p < 1,$ though, we don't get a discretely defined norm, because it violates the triangle inequality. The *triangle inequality* states that any given side of a triangle must be less than or equal to the sum of the other two sides.

Matrix Norms

A matrix norm is a function from $\mathbb{R}^{nxn} \to \mathbb{R}$ that satisfies a given number of properties, symbolized by $||A||$ given a matrix A.

The properties are as follows:

1. $\|A\| > 0$ *for all* $M \in \mathbb{R}^{nxn}$ *and also* $\|A\| = 0$ *if* $A = 0$
2. $\|aM\| = |a| * \|M\|$ *for all* $a \in \mathbb{R}^n$
3. $\|M + N\| \leq \|M\| + \|N\|$
4. $\|MN\| \leq \|M\| * \|N\|$

Inner Products

An important type of vector space that's referenced often in machine learning literature is the inner product. This element of vector space allows someone to know the length of a vector or the angle between two vectors. In addition, you can also determine from the inner product normed vector space. Specifically, the *inner product* is the function utilized in the kernels of support vector machines to compute the images of the data that the support vector machine puts into feature space from the input space. The inner product space of is a function ⟨.,.⟩ defined by the following, where u and v are vectors, $u = [u_1, u_2, \ldots, u_n], v = [v_1, v_2, \ldots, v_n]$:

$$\langle u,v \rangle = u_1 v_1 + u_2 v_2 + \ldots + u_n v_n \quad \textit{for } i = 1,2,\ldots,n$$

For a function to be an inner product, it must satisfy three axioms:
Conjugate symmetry:

$$\langle u,v \rangle = \langle v,u \rangle$$

Linearity in the first argument:

$$\langle au + bv, w \rangle = a\langle u,w \rangle + b\langle v,w \rangle$$

Positive-definiteness:

$$\textit{For any } u \in V, \langle u,u \rangle \geq 0; \textit{ and } \langle u,u \rangle = 0 \textit{ only if } u = 0$$

Norms on Inner Product Spaces

Inner product spaces naturally have a defined norm, which is based upon the norm of the space itself, given by the following:

$$\|\langle x \rangle\| = \sqrt{\langle x \rangle}$$

Directly from the axioms, we can prove the following: The Cauchy-Schwartz inequality states that for all vectors u and v of an inner product space, the following is true:

$$\|\langle u,v \rangle\|^2 \leq \langle u,u \rangle * \langle v,v \rangle$$

$$|\langle u,v \rangle| \leq \|u\| * \|v\|$$

The two sides are only considered equal if and only if u and v are linearly dependent, which means that they would have to be parallel, one of the vectors has a magnitude of zero, or one is a scalar multiplier of the other.

Proofs

First proof: expanding out the brackets and collecting together identical terms yields the following equation:

$$\Sigma_i^n \Sigma_j^n \left(a_i b_j - a_j b_i\right)^2 = \left(\Sigma_i^n a_i^2\right)\Sigma_j^n b_j^2 + \left(\Sigma_i^n b_i^2\right)\Sigma_j^n a_j^2 - 2\left(\Sigma\, a_i b_i\right)\Sigma_j^n b_j a_j$$

$$= 2\left(\Sigma_i^n a_i^2\right)\left(\Sigma_i^n b_i^2\right) - 2\left(\Sigma_i^n a_i b_i\right)^2$$

Because the lefthand side of the equation is the sum of squares of real numbers, it is greater than or equal to zero. As such, the following must be true:

$$\left(\Sigma_i^n a_i^2\right)\left(\Sigma_i^n b_i^2\right) \geq \left(\Sigma_i^n a_i b_i\right)^2$$

Second proof: consider the following quadratic polynomial equation:

$$f(x) = \left(\Sigma_i^n a_i^2\right)x^2 - 2\left(\Sigma_i^n a_i b_i\right)x + \Sigma_i^n b_i^2 = \Sigma\left(a_i x - b_i\right)^2$$

Because $f(x) \geq 0$ *for* any $x \in \mathbb{R}$, it follows that the discriminant of $f(x)$ is negative, and therefore the following must be the case:

$$\left(\Sigma_i^n a_i b_i\right)^2 - \left(\Sigma_i^n a_i^2\right)\left(\Sigma_i^n b_i^2\right) \leq 0$$

Third proof: consider the following two Euclidean norms A and B:

$$Let\ A = \sqrt{a_1^2 + a_2^2 + \ldots + a_n^2},\ \ B = \sqrt{b_1^2 + b_2^2 + \ldots + b_n^2}$$

By the arithmetic-geometric means inequality, we have

$$\frac{\Sigma_i^n\left(a_i b_i\right)}{AB} \leq \Sigma_i^n\left(\frac{1}{2}\right)\left(\left(\frac{a_i^2}{A^2}\right) + \left(\frac{b_i^2}{B^2}\right)\right) = 1,$$

such that

$$\Sigma\, a_i b_i \leq AB = \sqrt{a_1^2 + a_2^2 + \ldots + a_n^2}\,\sqrt{b_1^2 + b_2^2 + \ldots + b_n^2}$$

Thus, the following is yielded:

$$\left(\Sigma\, a_i b_i\right)^2 \leq \left(\Sigma_i^n\, a_i^2\right)\left(\Sigma_i^n\, b_i^2\right)$$

Orthogonality

Orthogonality is described as a measure or degree of unrelatedness. For example, an orthogonal transformation of a vector yields a vector such that it is unrelated to the vector we transformed. The geometric interpretation of the inner product in terms of angle and length motivates much of the terminology we use in regard to those spaces. Indeed, an immediate consequence of the Cauchy-Schwarz inequality is that it justifies defining the angle between two non-zero vectors:

$$\text{Angle}(\text{x,y}) = \arccos \frac{\langle x, y \rangle}{\|x\| * \|y\|}$$

Outer Product

The tensor product of two vectors is related slightly to the inner product previously defined. A *tensor* product is a way of creating a new vector space analogous to multiplication of integers:

$$\text{Let u and v equal two vectors where } \text{x} = [x_1, x_2, x_3],\ y = [y_1, y_2, y_3]^T$$

$$y \otimes x = yx^T = \begin{matrix} y_1 \\ y_2 \\ y_3 \end{matrix} * \begin{matrix} x_1 & x_2 & x_3 \end{matrix} = \begin{matrix} y_1x_1 & y_1x_2 & y_1x_3 \\ y_2x_1 & y_2x_2 & y_2x_3 \\ y_3x_1 & y_3x_2 & y_3x_3 \end{matrix}$$

Eigenvalues and Eigenvectors

An *eigenvalue* is a number derived from a square matrix, which corresponds to a specific *eigenvector*, also associated with a square matrix. Together, they "provide the eigendecomposition of a matrix." Plainly spoken, the eigendecomposition of a matrix merely provides the matrix in the form of eigenvectors and their corresponding eigenvalues. Eigendecomposition is important because it is a "method by which we can find the maximum (or minimum) of functions involving matrices."

Eigendecomposition:

$$Au = \lambda u$$

$$(A - \lambda I)u = 0$$

Where A = square matrix, and u = eigenvector to matrix A (if length of vector changes when multiplied by A):

$$\lambda = \text{eigenvalue to corresponding eigenvecvtor u}$$

Assume the following is also true:

$$A = \begin{pmatrix} 2 & 3 \\ 2 & 1 \end{pmatrix}$$

Therefore:

$$u_1 = \begin{pmatrix} 3 \\ \overline{2} \end{pmatrix},\ u_2 = \begin{pmatrix} -\frac{1}{1} \end{pmatrix},\ \lambda_1 = 4,\ \lambda_2 = -1$$

For most applications, the eigenvectors are normalized to a unit vector as such:

$$u^T u = 1$$

Eigenvectors of A furthermore are put together in a matrix U. Each column of U is an eigenvector of A. The eigenvalues are stored in a diagonal matrix ^, where the trace, or diagonal, of the matrix gives the eigenvalues. Thus we rewrite the first equation accordingly:

$$AU = UA$$

$$A = U \wedge U^{-1}$$

$$= \begin{bmatrix} 3 & -1 \\ 2 & 1 \end{bmatrix} \begin{bmatrix} 4 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} 2 & 2 \\ -4 & 6 \end{bmatrix}$$

$$= \begin{bmatrix} 2 & 3 \\ 2 & 1 \end{bmatrix}$$

A graphical representation of eigenvectors is given in Figure 2-6.