CSC461
INTRODUCTION TO DATA SCIENCE

Dr. Muhammad Sharjeel

https://muhmmadsharjeel.github.io/

07

RELATIONAL DATA, MATRICES, AND LINEAR ALGEBRA

RELATIONAL DATA

- The term “relation” can be interchanged with the standard notion of “tabular data”
- Tables are simply combinations of rows and columns
- Rows are called tuples (or records) that represent a single instance of a relation (table), and must be unique
- Columns are called attributes that specify some element contained by each of the tuples

- Example of a relation (or table)

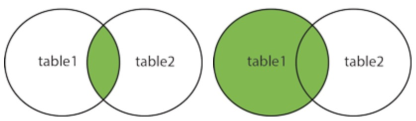
ID	First Name	Last Name	Role
1	Ali	Nawaz	Instructor
2	Nadeem	Ahmed	TA
3	Waqas	Khan	TA
4	Fatima	Gul	Student
5	Haroon	Mirza	Student

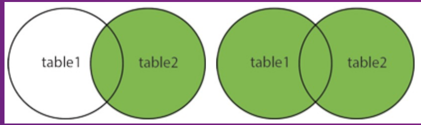
RELATIONAL DATA

- Primary key
  - Unique ID for every tuple in a relation (i.e., every row in the table)
  - Each relation (table) must have exactly one primary key
- Foreign key
  - Attribute that points to the primary key of another relation
  - Deletion of a primary key requires to delete all foreign keys pointing to it
- Indexes are created as ways to “quickly” access elements of a table
  - For example, finding someone with last name “Ahmed”
  - No option but search throughout the whole table, *O*(n) operation
- Index is a kind of a separate sorted table containing the indexed column and the tuple location
  - Searching for a value using indexing takes *O*(log n)
  - Primary key always has an index associated with it
- Indexes don’t have to be on a single column
  - Index(es) over multiple columns needs some ordering

RELATIONAL DATA

- Inter-table relationships, relate one (or more) rows in a table with one (or more) rows in another table, via a foreign key
- Several types of inter-table relationships
  - One-to-one
  - One-to-many
  - Many-to-many
- Join operations merge multiple tables into a single relation
  - Can then be saved as a new table or just directly used
- Four types of joins:
  - Inner
  - Left
  - Right
  - Outer





- Two tables are joined on columns from each table, where these columns specify which rows are kept

PANDAS

- Pandas is a “Data Frame” library in Python, meant for manipulating in-memory data with row and column labels (as opposed to, e.g., matrices, that have no row or column labels)
- Pandas
  - is not a relational database system, but contains functions that mirror some functionality of relational databases
  - has no notion of primary keys (but it does have indexes)
  - Operations are typically not in place
    - return a new modified DataFrame, rather than modifying an existing one
  - Use the “inplace” flag to make them done in place
- Selecting a single row or column in a Pandas DataFrame will return a “Series” object (a one-dimensional DataFrame)
  - It has only an index and corresponding values, not multiple columns

SQLITE

- SQLite is an actual relational database management system (RDBMS)
- Serverless model, applications directly connect to a file
- Allows simultaneous connections from many applications to the same database file
- All operations in SQLite use SQL (Structured Query Language) commands

VECTORS

- A vector is a *1D* array of values
- Notation $x \in \mathbb{R}^n$ is used to denote that x is an n-dimensional vector with real-valued entries

$$x = \begin{bmatrix} x_1, \\ x_2, \\ x_3, \\ . \\ . \\ x_n \end{bmatrix}$$

- Use the notation x_i to denote the *i*th entry of x
- Vectors (most commonly) represent column vectors, to consider a row vector, x^T notation is used

MATRICES

- Matrices are the most common way of representing data to be analyzed and manipulated by virtually any data science or analytics algorithm
- Matrices are
  - excellent choice to store tabular data
  - foundation of linear algebra
- A matrix is a *2D* array of values
- Notation $\mathbf{A} \in \mathbb{R}^{\mathbf{mxn}}$ is used to denote a real-valued matrix with *m* rows and *n* columns

$$A = \begin{bmatrix} A_{11} & A_{12} & \dots & A_{1n} \\ A_{21} & A_{22} & \dots & A_{2n} \\ A_{m1} & A_{m2} & \dots & A_{mn} \end{bmatrix}$$

- A_{ij} is the entry in row *i* and column *j*
- A_i refers to row *i*, A_j refers to column *j*

MATRICES

- Understanding both matrices and linear algebra is critical for virtually all data science algorithms
- Matrices store tabular data (particularly numerical entries) in an efficient manner
- Example:

ID	HW-1 Marks	HW-2 Marks
5	10	8
9	4	8
25	8	3

- A matrix could be used to represent this data (ignoring primary key)

$$A \in \mathbb{R}^{3x2} = \begin{bmatrix} 10 & 8 \\ 4 & 8 \\ 8 & 3 \end{bmatrix}$$

MATRICES

- Matrices can be laid out in memory by row or by column

$$A = \begin{bmatrix} 10 & 8 \\ 4 & 8 \\ 8 & 3 \end{bmatrix}$$

- Row major ordering: 10, 8, 4, 8, 8, 3
- Column major ordering: 10, 4, 8, 8, 8, 3
- Matrices that contain mostly zero values are called sparse
- Matrices where most of the values are non-zero are called dense

MATRICES

- Matrices and vectors also provide a way to express and analyze systems of linear equations
- Consider two linear equations (two unknowns, i.e., x_1 and x_2)

$$4x_1 - 5x_2 = -13$$
$$-2x_1 + 3x_2 = 9$$

- Using matrix notation, they can be written as:

$$Ax = B$$

$$A = \begin{bmatrix} 4 & -5 \\ -2 & 3 \end{bmatrix} \quad B = \begin{bmatrix} -13 \\ 9 \end{bmatrix} \quad x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

BASIC MATRIX OPERATIONS

- For two matrices A, B $\in \mathbb{R}^{mxn}$, matrix addition/subtraction is just the elementwise addition or subtraction of their elements

  $$C \in \mathbb{R}^{mxn} = A+B, \quad C_{ij} = A_{ij} + B_{ij}$$

- For A $\in \mathbb{R}^{mxn}$, transpose is an operator that “flips” rows and columns

  $$C \in \mathbb{R}^{mxn} = A^T, \quad C_{ji} = A_{ij}$$

- For A $\in \mathbb{R}^{mxn}$, B $\in \mathbb{R}^{nxp}$ matrix multiplication is defined as

  $$C \in \mathbb{R}^{mxp} = AB, \quad C_{ij} = \sum_{k=1}^{n} A_{ik} B_{kj}$$

- There is no concept of matrices division

BASIC MATRIX OPERATIONS

- Addition and Subtraction Examples:

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 7 & 8 & 9 \end{bmatrix} + B = \begin{bmatrix} 5 & 6 & 7 \\ 3 & 4 & 5 \end{bmatrix}$$

$$A + B = \begin{bmatrix} 6 & 8 & 10 \\ 10 & 12 & 14 \end{bmatrix}$$

$$A - B = \begin{bmatrix} -4 & -4 & -4 \\ 4 & 4 & 4 \end{bmatrix}$$

BASIC MATRIX OPERATIONS

- Multiplication Example:

$$A = \begin{bmatrix} 100 & 200 \\ 300 & 400 \end{bmatrix}$$

$$5A = 5 * \begin{bmatrix} 100 & 200 \\ 300 & 400 \end{bmatrix}$$

$$5A = \begin{bmatrix} 500 & 1000 \\ 1500 & 2000 \end{bmatrix}$$

BASIC MATRIX OPERATIONS

- Transpose Example:

$$A = \begin{bmatrix} 2 & -9 & 3 \\ 13 & 11 & 7 \\ 3 & 6 & 15 \\ 4 & 13 & 1 \end{bmatrix}$$

$$AT = \begin{bmatrix} 2 & 13 & 3 & 4 \\ -9 & 11 & 6 & 13 \\ 3 & 7 & 15 & 1 \end{bmatrix}$$

BASIC MATRIX OPERATIONS

- The identity matrix $I \in \mathbb{R}^{nxn}$, is a square matrix with ones on diagonal and zeros elsewhere

$$I = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

- For a square matrix $S \in \mathbb{R}^{nxn}$, inverse $S^{-1} \in \mathbb{R}^{nxn}$ such that

$$S = \begin{bmatrix} a & b \\ c & d \end{bmatrix}^{-1} = \begin{bmatrix} d & -b \\ -c & a \end{bmatrix}$$

LINEAR ALGEBRA

- Linear algebra computations underlie virtually all data science algorithms
  - In different computer languages, there has been massive efforts to write extremely fast linear algebra code
  - For example, multiplication of large matrices, specialized code (libraries) work a lot faster than standard ‘nested loops’
- In Python, the standard library for matrices, vectors, and linear algebra is **Numpy**
- It provides both a framework for storing tabular data as multidimensional arrays and linear algebra routines
- Numpy *ndarrays* are multi-dimensional arrays, routines that act like matrices or vectors
- Specialized Python libraries like BLAS (Basic Linear Algebra Subprograms) and LAPACK (Linear Algebra PACKage) provide interfaces for basic matrix multiplication (BLAS) and fancier linear algebra methods (LAPACK)
  - Highly optimized version of these libraries: ATLAS, OpenBLAS, Intel MKL

PANDAS TUTORIAL

- Pandas is essentially the data’s home
- It helps in cleaning, transforming, and analyzing the data
- It is built on top of the NumPy, so a lot of the structure of NumPy is used or replicated
- Two primary components of pandas are the **Series** and **DataFrame**
- Series is a column, and a DataFrame is a multi-dimensional table made up of a collection of Series
- Refer to iPython Notebook for detailed Pandas tutorial

THANKS