Heatmaps are used to visualize data in a tabular format where the cells are colored based on the values they contain. They are particularly useful for identifying patterns, trends, and correlations in large datasets.:

1. **Visualizing Relationships**
2. **Identifying Clusters**
3. **Correlation Analysis**
4. **Comparing Data Across Categories**
5. **Ease of Interpretation**
6. **Customization**

NumPy: Efficient Numerical Computation ,Array Operations,Integration with other Libraries

PANDAS: Data Manipulation and Analysis,Time Series Data,Missing Data Handling

header=None: This argument specifies whether the CSV file has a header row (i.e., column names). When header=None, Pandas will automatically generate integer column names (0, 1, 2, ...) for the DataFrame.

scikit-learn is a popular machine learning library in Python that provides a wide range of tools for machine learning tasks such as classification,regression, clustering, dimensionality reduction, and more.

LabelEncoder: It's commonly used to transform categorical labels into numerical labels, which can then be fed into machine learning models.For example, if you have a categorical feature like "red", "blue", "green", LabelEncoder can transform these into numerical values like 0, 1, 2, etc.

StandardScaler:It's commonly used to preprocess numerical input features before feeding them into machine learning models.

This parameter sets the random seed for reproducibility. Providing a specific value (e.g., random\_state=42) ensures that the data split is reproducible. If you want the split to be different each time you run the code, you can omit this parameter or provide a different seed value.

Convert all non-numeric columns to numeric (if needed)

X = pd.get\_dummies(X) Convert categorical variables to dummy/indicator variables.

sns.regplot(): This function overlays a regression line on the scatter plot. It fits a linear regression model to the data and plots the resulting line. The scatter=False parameter ensures that only the regression line is shown, not the individual data points.

Calculate Z-scores: For each data point in the dataset, calculate its Z-score, which measures how many standard deviations the data point is from the mean.

bins=10: This parameter specifies the number of bins (or intervals) into which the data will be divided. In this case, the data will be divided into 10 bins.

plt.subplot(1, 2, 1): This line of code sets up a subplot grid with 1 row and 2 columns, and selects the first subplot for the subsequent plot. Example in assign both male and female are shown in 1 row and 2 column.

plt.figure(figsize=(14, 6)): initializes a new Matplotlib figure with a specific size of 14 inches in width and 6 inches in height

jitter=True: This parameter adds a small amount of random jitter to the x-coordinate of each data point, which helps to prevent overlapping points and provides a better visualization of the distribution of data.

kde=True: This parameter adds a kernel density estimate (KDE) curve to the histogram, providing a smoothed estimate of the underlying probability density function of the data.

orient='v': This parameter sets the orientation of the box plot to vertical (i.e., the boxes will be drawn vertically).

Histogram and Scatter plot to Box Plot:

1. **Histogram**:
   * **Purpose**: Histograms are used to visualize the distribution of a single continuous variable.
   * **Representation**: In a histogram, the data is divided into bins, and the height of each bar represents the frequency or count of values falling within that bin.
   * **Insights**: Histograms help to understand the central tendency, spread, and shape of the data's distribution, including any peaks, gaps, or skewness.
   * **Example**: A histogram of ages might show that most individuals fall within a certain age range, with fewer individuals at the extremes.
2. **Scatter Plot**:
   * **Purpose**: Scatter plots are used to visualize the relationship between two continuous variables.
   * **Representation**: In a scatter plot, each data point is represented by a dot, and the position of the dot on the plot corresponds to the values of the two variables being compared.
   * **Insights**: Scatter plots help to understand patterns, trends, and correlations between variables. They can reveal linear or non-linear relationships, clusters, outliers, and the strength of associations.
   * **Example**: A scatter plot of age versus blood pressure might show a positive correlation, indicating that blood pressure tends to increase with age.
3. **Box Plot**:
   * **Purpose**: Box plots are used to visualize the distribution of a continuous variable within different categories or groups.
   * **Representation**: In a box plot, the central rectangle represents the interquartile range (IQR) of the data, with the median indicated by a horizontal line inside the box. The "whiskers" extend from the minimum to the maximum values of the data, excluding any outliers, which are represented as individual points outside the whiskers.
   * **Insights**: Box plots help to compare the central tendency, spread, and variability of a variable's distribution across different groups. They also provide information about potential outliers and the overall shape of the distribution within each group.
   * **Example**: A box plot comparing ages across different genders might show differences in the median age, variability, and presence of outliers between males and females.

Hadoop is an open-source framework designed to process and store large volumes of data in a distributed computing environment. It provides a scalable, reliable, and cost-effective solution for handling Big Data by distributing the data processing tasks across clusters of commodity hardware.

Key components of the Hadoop ecosystem include:

1. **Hadoop Distributed File System (HDFS)**: A distributed file system that stores data across multiple machines in a Hadoop cluster. HDFS provides high-throughput access to application data and is designed to handle large files efficiently.
2. **MapReduce**: A programming model and processing framework for parallel data processing. MapReduce divides data processing tasks into smaller, independent tasks that can be executed in parallel across multiple nodes in a Hadoop cluster. It consists of two main phases: the Map phase, where data is processed and transformed into intermediate key-value pairs, and the Reduce phase, where intermediate results are aggregated and processed to produce the final output.
3. **YARN (Yet Another Resource Negotiator)**: A resource management and job scheduling framework in Hadoop. YARN manages resources (CPU, memory, etc.) across the cluster and schedules jobs to run on available resources. It enables multiple data processing frameworks to run concurrently on the same Hadoop cluster, allowing for more efficient resource utilization.
4. **Hadoop Common**: A set of utilities and libraries used by other Hadoop components. Hadoop Common provides the necessary infrastructure and tools for managing and interacting with Hadoop clusters.
5. **Hadoop ecosystem projects**: In addition to the core components mentioned above, the Hadoop ecosystem includes various projects and tools that extend the capabilities of Hadoop for specific use cases. Some popular ecosystem projects include Apache Hive (data warehousing), Apache Pig (data processing), Apache Spark (in-memory processing), Apache HBase (NoSQL database), Apache Kafka (stream processing), and Apache Sqoop (data transfer between Hadoop and relational databases), among others.
6. The **TokenizerMapper** class extends the **Mapper** class provided by the Hadoop MapReduce framework. This class is responsible for processing input data and emitting intermediate key-value pairs.
7. The **IntSumReducer** class extends the **Reducer** class provided by the Hadoop MapReduce framework. This class is responsible for aggregating intermediate key-value pairs and producing the final output.

* Facebook developed its own distributed storage system called "Haystack" to store photos efficiently.
* Facebook also utilizes Apache HBase for real-time analytics and other data-intensive tasks.

Instagram uses some form of Hadoop or similar distributed processing systems for handling analytics and processing large datasets.

| **Feature** | **HDFS** | **Google File System (GFS)** |
| --- | --- | --- |
| Development | Part of Apache Hadoop | Developed by Google |
| Hardware | Commodity | - |
| Workload | Batch processing | Various applications and services |
| Architecture | Master-slave | Master-slave |
| Components | NameNode, DataNodes | Master Server, Chunkservers |
| Replication | Yes | Yes |
| Data Locality | Yes | - |
| Integration | Strong integration with Hadoop ecosystem | - |
| Atomic Record Appends | - | Yes |
| Automatic Replication | - | Yes |
| Snapshot Creation | - | Yes |

| **Feature** | **Hive** | **HBase** |
| --- | --- | --- |
| Type | Data warehousing tool | NoSQL distributed database |
| Query Language | HiveQL (SQL-like) | HBase Shell (Java API) |
| Use Case | Batch processing, SQL queries | Real-time read/write operations |
| Storage Format | Tables with schema | Sparse, structured or semi-structured data |
| Data Model | Table-based | Column-family based |
| Indexing | Indexes supported | Automatic sharding |
| Transactions | ACID compliant (with certain configurations) | ACID compliant (with certain configurations) |
| Scalability | Good for analytical queries | Good for random reads/writes |
| Latency | High latency for individual record reads | Low latency for individual record reads |
| Compression | Supported | Supported |
| Secondary Indexes | Supported (in recent versions) | Supported |
| Data Size | Suitable for large datasets | Suitable for large datasets |

1. **Hadoop Common:** This includes libraries and utilities used by other Hadoop modules. For example, it provides the Java Archive (JAR) files and scripts needed to start Hadoop services.
2. **Hadoop Distributed File System (HDFS):** This is Hadoop's file system designed for storing large files across multiple machines. An example is storing log files from multiple servers in a Hadoop cluster to perform analytics on them collectively.
3. **MapReduce:** It's a programming model for processing large datasets in parallel across a distributed cluster. An example would be analyzing a large dataset of sales transactions to calculate total sales by region using map and reduce functions.
4. **YARN (Yet Another Resource Negotiator):** YARN is a resource management layer responsible for managing resources and scheduling applications. An example is running multiple MapReduce jobs simultaneously on a Hadoop cluster without resource contention.
5. **HBase:** It's a NoSQL database that provides real-time read/write access to large datasets. An example is storing user profiles and activity logs in HBase for a social media platform like Twitter to quickly retrieve user data.
6. **Hive:** A data warehouse infrastructure built on Hadoop, which provides SQL-like queries for data analysis. An example is using Hive to query and analyze website traffic logs stored in HDFS to understand user behavior.
7. **Pig:** It's a platform for analyzing large datasets using a high-level scripting language called Pig Latin. An example is using Pig to clean and transform log files before loading them into HDFS.
8. **Sqoop:** Sqoop is used to efficiently transfer data between Hadoop and relational databases. An example is importing data from a MySQL database into HDFS for analysis using MapReduce.
9. **Flume:** It's used for efficiently collecting, aggregating, and moving large amounts of log data. An example is using Flume to collect log data from web servers and store it in HDFS for analysis.
10. **Oozie:** Oozie is a workflow scheduler for managing Hadoop jobs. An example is scheduling a series of MapReduce jobs and Hive queries to run sequentially to process and analyze a dataset.
11. **Mahout:** Mahout is a machine learning library built on Hadoop for implementing algorithms such as collaborative filtering and clustering. An example is using Mahout to build a recommendation engine for an e-commerce website.
12. **ZooKeeper:** ZooKeeper is a centralized service for maintaining configuration information and providing distributed synchronization. An example is using ZooKeeper to coordinate distributed applications running on a Hadoop cluster.

**Cloudera:**

* Hadoop distribution
* Core Hadoop ecosystem components
* Cloudera Manager
* Cloudera Navigator
* Simplifies deployment and management
* Scalable
* Enterprise-grade support

1. **Data Storage:**
   * **Hive:** Hive is primarily used for structured data storage and querying. It stores data in tables with a predefined schema, similar to traditional relational databases. Hive tables are typically stored in HDFS, but they can also be stored in other Hadoop-compatible file systems.
   * **HBase:** HBase, on the other hand, is a NoSQL database that is optimized for storing and retrieving semi-structured or unstructured data. It provides real-time random read/write access to large datasets and is suitable for scenarios requiring fast data retrieval based on key-value lookups.
2. **Data Model:**
   * **Hive:** Hive follows a table-based data model, where data is organized into tables with rows and columns. It supports SQL-like querying through HiveQL.
   * **HBase:** HBase follows a column-family-based data model, where data is stored in key-value pairs within column families. It provides flexible schema design and is well-suited for sparse data.
3. **Querying:**
   * **Hive:** HiveQL, a SQL-like query language, is used to query data stored in Hive tables. It translates SQL queries into MapReduce or Tez jobs for execution on the Hadoop cluster.
   * **HBase:** HBase provides a Java API for data manipulation. Queries in HBase are typically key-based lookups or range scans, suitable for real-time access to individual records.
4. **Use Cases:**
   * **Hive:** Hive is commonly used for batch processing, data warehousing, and analytical queries over large datasets. It is suitable for scenarios where data is structured and analytical insights are required.
   * **HBase:** HBase is used for real-time applications requiring fast data access and updates. It is suitable for scenarios where data is semi-structured or unstructured, and low-latency access is critical, such as in web applications, sensor data processing, and time-series data analysis.
5. **Overfitting:**
   * **Definition:** Overfitting occurs when a model learns the training data too well, capturing noise or random fluctuations in the data rather than the underlying pattern. As a result, the model performs well on the training data but fails to generalize to new, unseen data.
   * **Characteristics:**
     + High accuracy on training data.
     + Poor performance on validation or test data.
     + Model captures noise or irrelevant patterns in the data.
   * **Causes:**
     + Complex models with too many parameters relative to the amount of training data.
     + Insufficient regularization.
     + Noisy or redundant features.
   * **Prevention/Remedies:**
     + Use simpler models with fewer parameters.
     + Add regularization techniques such as L1/L2 regularization, dropout, or early stopping.
     + Increase the amount of training data.
     + Perform feature selection or dimensionality reduction.
6. **Underfitting:**
   * **Definition:** Underfitting occurs when a model is too simple to capture the underlying structure of the data. The model performs poorly on both the training and validation/test data because it fails to learn the patterns present in the data.
   * **Characteristics:**
     + Low accuracy on both training and validation/test data.
     + Model fails to capture the underlying pattern in the data.
   * **Causes:**
     + Models are too simple or have insufficient capacity to capture the complexity of the data.
     + Insufficient training time or inadequate optimization.
     + Missing relevant features or inadequate feature engineering.
   * **Prevention/Remedies:**
     + Use more complex models with higher capacity.
     + Increase the amount of training data.
     + Improve feature engineering to include relevant features.
     + Train the model for longer or adjust hyperparameters to improve optimization.