Domain 1: Data Processing

1.1 Collect, Ingest, and Store Data

1.1.1 COLLECT DATA

High-Performing data

- ***REPRESENTATIVE***
  - ***Best practice**: When building an ML model, it's important to feed it high-quality data that accurately reflects the real world. For example, if 20 percent of customers typically cancel memberships after a year, the data should represent that churn rate. Otherwise, the model could falsely predict that significantly more or fewer customers will cancel.*
  - ***Watch for**: If your data doesn't actually reflect the real-world scenarios that you want your model to handle, it will be difficult to identify meaningful patterns and make accurate predictions.*
- ***RELEVANT***
  - ***Best practice**: Data should contain relevant attributes that expose patterns related to what you want to predict, such as membership duration for predicting cancellation rate.*
  - ***Watch for**: If irrelevant information is mixed in with useful data, it can impact the model's ability to focus on what really matters. For example, a list of customer emails in a dataset that's supposed to predict membership cancellation can negatively impact the model's performance.*
- ***Feature Rich***
  - ***Best practice**: Data should include a complete set of features that can help the model learn underlying patterns. You can identify additional trends or patterns to increase accuracy by including as much relevant data as possible.*
  - ***Watch for**: Data that has limited features reduces the ability of the ML algorithm to accurately predict customer churn. For example, if the data consists of a small set of customer details and omits important data, like demographic information, it will lose accuracy and miss opportunities for detecting patterns in cancellation rate.*
- ***Consistent***
  - ***Best practice**: Data must be consistent when it comes to attributes, such as features and formatting. Consistent data provides more accurate and reliable results.*
  - ***Watch for**: If the datasets come from various data sources that contain different formatting methods or metadata, the inconsistencies will impact the algorithm's ability to effectively process the data. The algorithm will be less accurate with the inconsistent data.*

Types of Data

Text

Text data, such as documents and website content, is converted to numbers for use in ML models, especially for natural language processing (NLP) tasks like sentiment analysis. Models use this numerical representation of text to analyze the data.

Tabular

Tabular data refers to information that is organized into a table structure with rows and columns, such as the data in spreadsheets and databases. Tabular data is ideal for linear regression models.

Time series

Time-series data is collected over time with an inherent ordering that is associated with data points. It can be associated with sensor, weather, or financial data, such as stock prices. It is frequently used to detect trends. For instance, you might analyze and forecast changes using ML models to make predictions based on historical data patterns.

Image

Image data refers to the actual pixel values that make up a digital image. It is the raw data that represents the colors and intensities of each pixel in the image. Image data, like data from photos, videos, and medical scans, is frequently used in machine learning for object recognition, autonomous driving, and image classification.

Formatting data

- ***Structured***
- ***Unstructured***
- ***Semi-structured***

Data formats and file types

1. Row-based data format

- *common in relational databases and spreadsheets.*
- *It shows the relationships between features*

customerID	name	age	email	last_support	subscription_ active
123456789	Rosalez, Alejandro	32	alejandro_rosalez@ example.com	1/11/22	false
87654321	Candella, Pat	22	pat_candella@ example.com	3/26/24	true

Row-based file types

- ***CSV***

Comma-separated values (CSV) files are lightweight, space-efficient text files that represent tabular data. Each line is a row of data, and the column values are separated by commas. The simple CSV format can store different data types like text and numbers, which makes it often used for ML data. However, the simplicity of CSV comes at the cost of performance and efficiency compared to columnar data formats with more optimized for analytics.

- ***Avro RecordIO***

Avro RecordIO is a row-based data storage format that stores records sequentially. This sequential storage benefits ML workloads that need to iterate over the full dataset multiple times during model training. Additionally, Avro RecordIO defines a schema that structures the data. This schema improves data processing speeds and provides better data management compared to schema-less formats.

2. Column-based data format

- *format, queries extract insights from patterns within a column rather than the entire record, which results in efficient analysis of trends across large datasets.*

Column-based file types

- ***Parquet***

Parquet is a columnar storage format typically used in analytics and data warehouse workloads that involve large data sets. ML workloads benefit from columnar storage because data can be compressed, which improves both storage space and performance.

- ***ORC***

Optimized row columnar (ORC) is a columnar data format similar to Parquet. ORC is typically used in big data workloads, such as Apache Hive and Spark. With the columnar format, you can efficiently compress data and improve performance. These performance benefits make ORC a widely chosen data format for ML workloads.

3. Object-notation data

- *Object notation fits non-tabular, hierarchical data, such as graphs or textual data.*
- *Object-notation data is structured into hierarchical objects with features and key-value pairs rather than rows and columns.*

Object-based file types

- ***JSON***

JavaScript Object Notation (JSON) is a document-based data format that is both human and machine readable. ML models can learn from JSON because it has a flexible data structure. The data is compact, hierarchical, and easy to parse, which makes it suitable for many ML workloads.

JSON is represented in objects and arrays.

<table>
<tr>
<td>An object is data defined by key-value pairs and enclosed in braces {}. The data can be a string, number, Boolean, array, object, or null</td>
<td>An array is a collection of values enclosed in square brackets [ ] and can contain values that are separated by commas. The following array consists of multiple objects.</td>
</tr>
<tr>
<td>{
"customerID": 123456789,
"name": "Rosalez, Alejandro",
"age": 32,
"email": "alejandro_rosalez@example.com",
"lastSupportInteraction": "1/11/22",
"subscriptionActive": false
}</td>
<td>[
{
"customerID": 123456789,
"name": "Rosalez, Alejandro",
"age": 32,
"email": "alejandro_rosalez@example.com",
"lastSupportInteraction": "1/11/22",
"subscriptionActive": false
},
{
"customerID": 87654321,
"name": "Candella, Pat",
"age": 22,
"email": "pat_candella@example.com",
"lastSupportInteraction": "3/26/24",
"subscriptionActive": true
}
]</td>
</tr>
</table>

- ***JSONL***

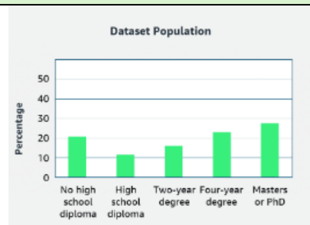
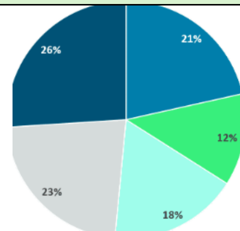
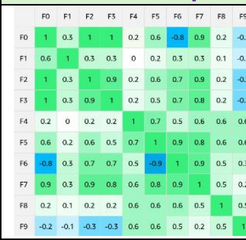
JavaScript Object Notation Lines (JSONL) is also called newline-delimited JSON. It is a format for encoding JSON objects that are separated by new lines instead of being nested. Each JSON object is written on its own line, such as in the following example.

```
{“customerID”: “12345678”. “name”: “Rosalez, Alejandro”, “age”: “32”, “email”:
“alejandro_rosalez@example.com”, “last_support”: “1/12/22”, “subscription_active”:
“false”}
{“customerID”: “87654321”. “name”: “Candella, Pat”, “age”: “22”, “email”:
“pat_candella@example.com”, “last_support”: “3/26/24“,“subscription_active”: “true”}
```

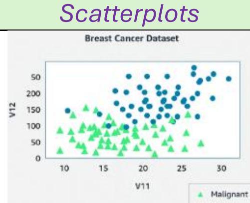
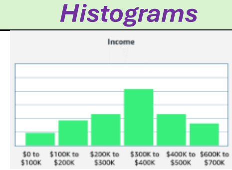
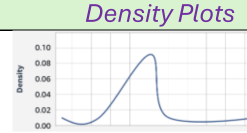
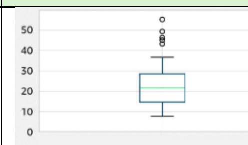
JSONL improves efficiency because individual objects can be processed without loading a larger JSON array. This improved efficiency when parsing objects results in better handling of large datasets for ML workloads. Additionally, JSONL structure can map to columnar formats like Parquet, which provides the additional benefits of those file types.

1.1.4 Graphs for data visualization

Categorical data

	Bar Charts	*Pie Charts*	*Heat maps*
Graphs			
Used for	***Comparison analysis***	***Composition analysis***	***Relationship analysis***
Features	*proportion of a dataset for specific attributes.*	*Entirety of the dataset.*	*use color to depict patterns and relationships*

Numerical Data

	Scatterplots	***Histograms***	*Density Plots*	*Box Plots*
Grapgh				
Features		*Data divided into bins*	*Similar to histograms,* • *smooth the distribution of data* • *don't constrain the data to bins.*	*displaying the location of key data points, such as median, quartiles, and outliers*
Assists with	***Relationship*** *analysis*	*Distribution analysis*	*Distribution analysis*	*Distribution analysis*
Useful for	*identify distinct regions*	*overall behavior of a single feature*	• *distribution of a single feature* • *No bins but continuous distribution*	• *quickly comparing distributions* • *identifying skewness, spread, and outliers.*

1.1.2 STORE DATA

Data Storage Options

1. ***S3***

   ***Features**: S3 serves as a central data lake for ingesting, extracting, and transforming data to and from other AWS services used for processing tasks. These tasks are an integral part of most ML workloads. The ability to store and retrieve data from anywhere makes Amazon S3 a key component in workflows requiring scalable, durable, and secure data storage and management.*

   ***Considerations**: S3 provides scalability, durability, and low cost, but it has higher latency compared to local storage. For latency-sensitive workloads, S3 might not be optimal. When deciding if S3 meets your needs, weigh its benefits against potential higher latency. With caching and proper architecture, many applications achieve excellent performance with S3, but you must account for its network-based nature.*

   Use cases

   - ***Data ingestion and storage***
     S3 can be used to store large datasets required for ML. Data can be ingested into S3 through streaming or batch processing. The data in S3 can then be used for ML training and inference. The scalability and durability of S3 makes it well-suited for storing the large volumes of data for effective machine learning.
   - ***Model training and evaluation***
     S3 stores ML datasets and models. It provides versioning to manage different model iterations, so you can store training and validation data in S3. You can also store trained ML models. With versioning, you can manage and compare models to evaluate performance.

   - ***Integration with other AWS services***
     S3 serves as a centralized location for other AWS services to access data. For example,

     - *SageMaker can access Amazon S3 data to train and deploy ML models.*
     - *Kinesis can stream data into S3 buckets for ingestion.*
     - *AWS Glue can connect to data stored in S3 for data processing purposes.*

2. ***EBS***

   ***Features**: Amazon EBS is well-suited for databases, web applications, analytics, and ML workloads. The service integrates with Amazon SageMaker as a core component for ML model training and deployment. By attaching EBS volumes directly to Amazon EC2 instances, you can optimize storage for ML and other data-intensive workloads.*

   ***Considerations**: EBS separates storage from EC2 instances, requiring more planning to allocate and scale volumes across instances. Instance stores simplify storage by tying storage directly to the EC2 instance lifecycle. This helps to avoid separate volume management. Although EBS offers flexibility, instance stores provide streamlined, intrinsic storage management than EBS.*

   Use cases

   - ***High-performance storage***
     EBS provides high-performance storage for ML applications requiring fast access to large datasets. EBS offers volumes with high IOPS for quick data reads and writes. The high throughput and IOPS accelerate ML workflows and applications.
   - ***Host pre-trained models***
     With EBS, you can upload, store, and access pre-trained ML models to generate real-time predictions without setting up separate hosting infrastructure.

3. ***EFS***

Features:

- *The service is designed to grow and shrink automatically as files are added or removed, so performance remains high even as file system usage changes.*
- *EFS uses the NFSv4 networking protocol to allow compute instances access to the file system across a standard file system interface. You can conveniently migrate existing applications relying upon on-premises NFS servers to Amazon EFS without any code changes.*

Considerations: *EFS has higher pricing, but offers streamlined scaling of shared file systems. EBS offers lower costs, but there are potential performance limitations based on workload. Consider if the higher EFS costs and potential performance variability are acceptable trade-offs compared to potentially lower EBS costs but workload-dependent performance excellent performance with S3, but you must account for its network-based nature.*

Use cases

- ***Concurrent access***
  EFS allows multiple EC2 instances to access the same datasets simultaneously. This concurrent access makes Amazon EFS well-suited for ML workflows that require shared datasets across multiple compute instances.
- ***Shared datasets***
  EFS provides a scalable, shared file system in the cloud that eliminates the need for you to copy large datasets to each compute instance. Multiple instances can access data, such as ML learning libraries, frameworks, and models, simultaneously without contention. This feature contributes to faster model training and deployment of ML applications.

4. Amazon FSx

Features:

- *Amazon FSx offers a rich set of features focused on reliability, security, and scalability to support ML, analytics, and high-performance computing applications.*
- *The service delivers millions of IOPS with sub-millisecond latency so you can build high-performance applications that require a scalable and durable file system.*

Considerations: *When using Amazon FSx for ML workloads, consider potential tradeoffs. Certain file system types and workloads can increase complexity and management needs. Tightly coupling the ML workflow to a specific file system also risks vendor lock-in, limiting future flexibility.*

Use cases

- ***Two types of file systems***
  FSx is a fully managed service that provides two types of file systems: Lustre and Windows File Server. Lustre allow for high-performance workloads requiring fast storage, such as ML training datasets.

- ***Distributed architecture***
  Lustre's distributed architecture provides highly parallel and scalable data access, making it ideal for hosting large, high-throughput datasets used for ML model training. By managing infrastructure operations, including backups, scaling, high availability, and security, you can focus on your data and applications rather than infrastructure management.

Model output Storage Options

1. Training Workloads

Training workloads require high performance and frequent random I/O access to data.

- *EBS volumes are well-suited for providing the random IOPS that training workloads need. Additionally, Amazon EBS instance store volumes offer extremely low-latency data access. This is because data is stored directly on the instances themselves rather than on network-attached volumes.*

2. Inference Workloads

Need fast response times for delivering predictions, but usually don't require high I/O performance, except for real-time inference cases.

- *EBS gp3 volumes or EFS storage options are well-suited for meeting these needs.*
- *For increased low-latency demands, upgrading to EBS io2 volumes can provide improved low-latency capabilities.*

3. Real-time and streaming workloads

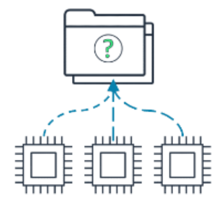
- *EFS file systems allow low latency and concurrent data access for real-time and streaming workloads. By sharing the same dataset across multiple EC2 instances, EFS provides high throughput access that meets the needs of applications requiring real-time data sharing.*

4. Dataset storage

- *S3 can be used for storing large datasets that do not need quick access, such as pretrained ML models, or data that is static or meant for archival purposes.*

Data Access Patterns

There are three common data access patterns in ML: copy and load, sequential streaming, and randomized access.

Copy and load *Data is copied from S3 to a training instance backed by EBS.*	***Sequential streaming*** *Data is streamed to instances as batches or individual records, typically from S3 to instances backed by EBS volumes.*	***Randomized access*** *Data is randomly accessed, such as with a shared file system data store, like FSx and EFS.*
		

Cost

Cost comparison

- ***S3*** *has the lowest cost for each gigabyte of storage based on storage classes. Storage classes are priced for each gigabyte, frequency of access, durability levels, and for each request.*
- ***EBS*** *has network attached storage, which is more expensive per gigabyte than Amazon S3. However, it provides lower latency, storage snapshots, and additional performance features that might be useful for ML workloads.*
- ***EFS*** *is a managed file service with increased costs that can link multiple instances to a shared dataset. Cost structure is designed around read and write access and the amount of gigabyte used with different storage tiers available.*
- ***FSx*** *pricing depends on the file system used. General price structure is around storage type used for each gigabyte, throughput capacity provisioned, and requests.*

AWS Tools for Reporting and Cost Optimization

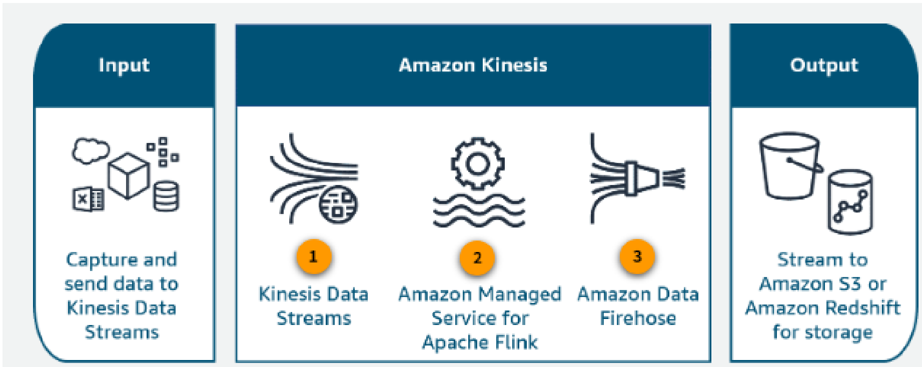
AWS provides several reporting and cost-optimization tools:

- *AWS Cost Explorer – See patterns in AWS spending over time, project future costs, identify areas that need further inquiry, observe Reserved Instance utilization, observe Reserved Instance coverage, and receive Reserved Instance recommendations.*
- *AWS Trusted Advisor – Get real-time identification of potential areas for optimization.*
- *AWS Budgets – Set custom budgets that trigger alerts when cost or usage exceed (or are forecasted to exceed) a budgeted amount. Budgets can be set based on tags and accounts as well as resource types.*
- *CloudWatch – Collect and track metrics, monitor log files, set alarms, and automatically react to changes in AWS resources.*
- *AWS CloudTrail – Log, continuously monitor, and retain account activity related to actions across AWS infrastructure at low cost.*
- *S3 Analytics – Automated analysis and visualization of S3 storage patterns to help you decide when to shift data to a different storage class.*
- *AWS Cost and Usage Report – Granular raw data files detailing your hourly AWS usage across accounts used for Do-It-Yourself (DIY) analysis (e.g., determining which S3 bucket is driving data transfer spend). The AWS Cost and Usage Report has dynamic columns that populate depending on the services you use.*

1.1.3 Data Ingestion

Realtime Ingestion - streaming services

Amazon Kinesis vs MSK vs Firehose



- ***Kinesis Data Streams*** *is primarily used for ingesting and processing data.*
- ***Firehose*** *provides a streamlined method of streaming data to data storage locations.*
- ***Amazon Managed Service for Apache Flink*** *provides consumption of streaming data using Apache Kafka in real-time for analysis.*

Streaming use cases

Data ingestion
Use Kinesis Data Streams for streaming real-time data from streaming data sources to data consumers. You can use Firehose for streaming data to a data repository, such as Amazon S3.

Data processing
Use Amazon Managed Service for Apache Flink to perform real-time processing, transformations, and feature engineering on data.

Real-time inference
Stream data processed by Amazon Managed Service for Apache Flink in real-time for machine learning processing to destinations, such as an Amazon SageMaker endpoint.

Data Extraction

Extraction

- ***Amazon S3 Transfer Acceleration***

Amazon S3 Transfer Acceleration uses CloudFront edge locations to accelerate large data transfers to and from S3. These transfers can help speed up data collection for ML workloads that require moving large datasets. S3 Transfer Acceleration overcomes bottlenecks like internet bandwidth and distance that can limit transfer speeds when working with large amounts of data.

- ***DMS***

AWS Data Migration Service (AWS DMS) facilitates database migration between databases or to Amazon S3 by extracting data in various formats, such as SQL, JSON, CSV, and XML. Migrations can run on schedules or in response to events for frequent data extraction. With AWS DMS, you can migrate databases between many sources and targets.

- ***AWS DataSync***

With AWS DataSync, you can efficiently transfer data between on-premises systems or AWS services by extracting data from sources, such as data file systems or network-attached storage. You can then upload data to AWS services like Amazon S3, Amazon EFS, Amazon FSx, or Amazon RDS on a scheduled or event-driven basis. DataSync facilitates moving large datasets to the cloud while reducing network costs and data transfer times.

- ***AWS Snowball***

AWS Snowball is a physical device service used to transfer large amounts of data into and out of AWS when network transfers are infeasible. Snowball devices efficiently and cost-effectively move terabytes or petabytes of data into S3

Storage

- ***S3***

With S3 serving as a highly scalable object storage service, data used for ML projects can be spread out across storage locations. Storage can be extracted and transferred to and from S3 with other AWS services. These other services include Amazon S3 Transfer Accelerator, AWS CLI, AWS SDK, AWS Snowball, AWS DataSync, AWS DMS, AWS Glue, and AWS Lambda.

- ***EBS***

EBS volumes provide storage for ML data. This data can be copied to services such as Amazon S3 or Amazon SageMaker, using tools like the AWS Management Console, AWS CLI, or AWS SDKs to manage volumes. EBS volumes store the necessary data that is then extracted and moved to other AWS services to meet ML requirements.

- ***EFS***

*Amazon EFS allows creating shared file systems that can be accessed from multiple EC2 instances, so you can share data across compute resources. You can extract data from **EFS** using AWS CLI, AWS SDKs, or with services like AWS Transfer Family and DataSync that facilitate data transfers. Amazon EFS provides the capability to share data from Amazon EC2 instances while also providing tools to conveniently move the data to other services.*

- ***RDS***

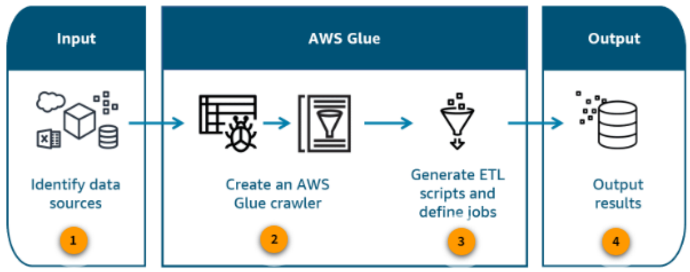
Amazon Relational Database Service (Amazon RDS) provides relational databases that can be accessed through AWS services like AWS DMS, the AWS CLI, and AWS SDKs to extract and transfer data. Amazon RDS is a common source for extracting relational data because it offers managed database instances that streamline data access.

- ***DynamoDB***

Amazon DynamoDB is a fully managed NoSQL database service provided by AWS. You can extract data using various AWS services like AWS DMS, AWS Glue, and AWS Lambda. You can use programmatic tools, such as the AWS CLI and AWS SDK, to process and analyze the data outside of DynamoDB. Data extraction allows DynamoDB data to be integrated with other platforms for further processing.

Data Merging

1. ***AWS Glue is a fully managed ETL service that you can use to prepare data for analytics and machine learning workflows.***



Best for: Glue works for ETL workloads from varied data sources into data lakes like Amazon S3.

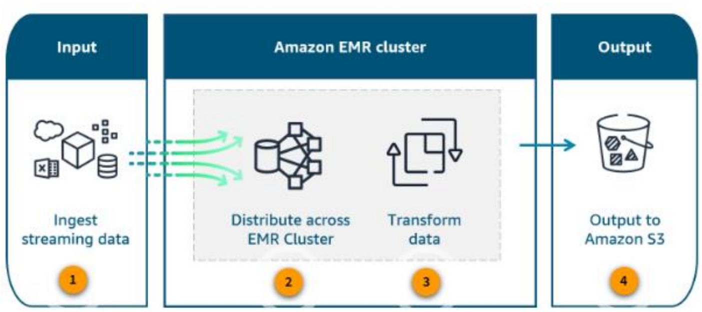
Steps

a) ***Identify: Identify data sources:*** *AWS Glue can be used to combine or transform large datasets using Apache Spark. It can efficiently process large structured and unstructured datasets in parallel. AWS Glue integrates with services like S3, Redshift, Athena, or other JDBC compliant data stores.*
b) ***Create an AWS Glue crawler:*** *AWS Glue crawlers scan data and populate the AWS Glue Catalog.*
c) ***Generate ETL scripts and define jobs:*** *Jobs run the ETL scripts to extract, transform, and load the data, which can start on demand or can be scheduled to run at specific intervals.*
d) ***Clean and transformed data*** *is written back to S3 or to another data store, such as Amazon Redshift.*

2. ***Amazon EMR***

Amazon EMR is a service for processing and analyzing large datasets using open-source tools of big data analysis, such as Apache Spark and Apache Hadoop. It applies ETL methodologies to ensure the product is flexible and scalable. Amazon EMR integrates data from multiple sources into one refined platform, making the transformation of data cost-effective and quick.

Best for: Amazon EMR is best suited for processing huge datasets in the petabyte range.



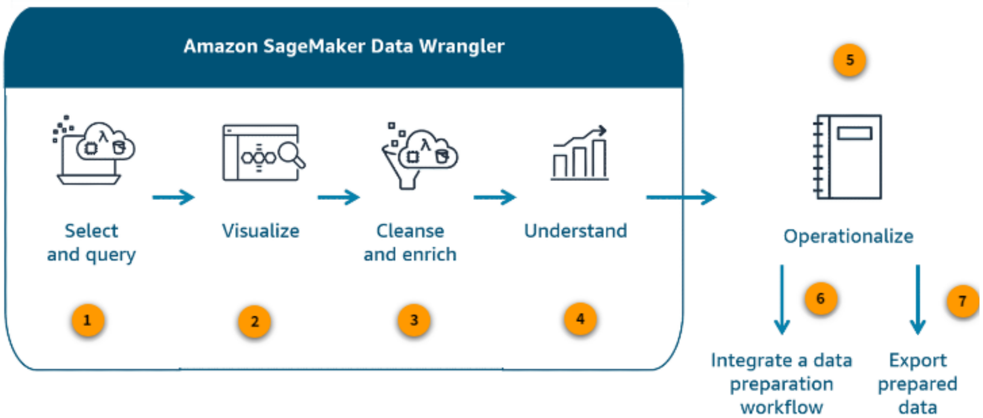
STEPS

- ***Ingest streaming sources:*** *ETL is done using Apache Spark Streaming APIs. This makes it possible to source data in real time from places such as Apache Kafka and Amazon Kinesis. Data is received and combined in real time.*
- ***Distribute across EMR cluster:*** *EMR clusters are made up of various nodes, each of which are configured specifically to handle parallel tasks and data processing.*
- ***Generate ETL scripts and define jobs:*** *At the end of the data processing lifecycle, you can use the Python, Scala, or SQL development environments. These environments give you powerful, flexible methods for scripting data workflows and for making data filtering, transformation, and aggregation more convenient.*

- **Output to Amazon S3:** *After processing and transforming the data, the results are outputted in an Amazon S3 bucket.*

- ***Amazon SageMaker Data Wrangler***

  Amazon EMR is a service for processing and analyzing large datasets using open-source tools of big data analysis, such as Apache Spark and Apache Hadoop. It applies ETL methodologies to ensure the product is flexible and scalable. Amazon EMR integrates data from multiple sources into one refined platform, making the transformation of data cost-effective and quick.



3.Clean and enrich
Cleanse and explore data, perform feature engineering with built-in data transforms, and detect statistical bias with Amazon SageMaker Clarify.
6.Integrate a data preparation workflow
Use Amazon SageMaker Pipelines to integrate a data preparation workflow.
7.Export prepared data
Export data to SageMaker Feature Store or S3.

When to use which (Data Wrangler vs Glue vs EMR)

Feature	AWS Glue	Amazon EMR	SageMaker Data Wrangler
Purpose	Serverless ETL service	Big data processing platform	ML-focused data preparation
Ease of Use	Visual ETL designer	Requires cluster setup and management	Visual interface, no coding required
Ideal Vol.	• Medium to large datasets	• Ideal for very large datasets	• Small to medium datasets
ML Integration	Can prepare data for ML, but not specialized	Can run ML frameworks, but requires setup	Tightly integrated with SageMaker ML workflow
Ideal Use Cases	• Batch data transformations • Data catalog management • Serverless data integration	• Batch and real-time processing • Complex big data processing	• Quick data exploration and visualization • Data cleaning and transformation for ML models

Troubleshooting

Scalability issues

- ***Capacity issues with data destinations***

With EFS, FSx, and S3, you can seamlessly scale storage increasing or decreasing in size.

- ***Latency issues, IOPs, and data transfer times***

High latency, insufficient IOPs, or slow data transfer times significantly impact performance of storage systems and data ingestion. Network bottlenecks, undersized provisioned storage volumes, or using inefficient methods of ingestion can arise from these factors.

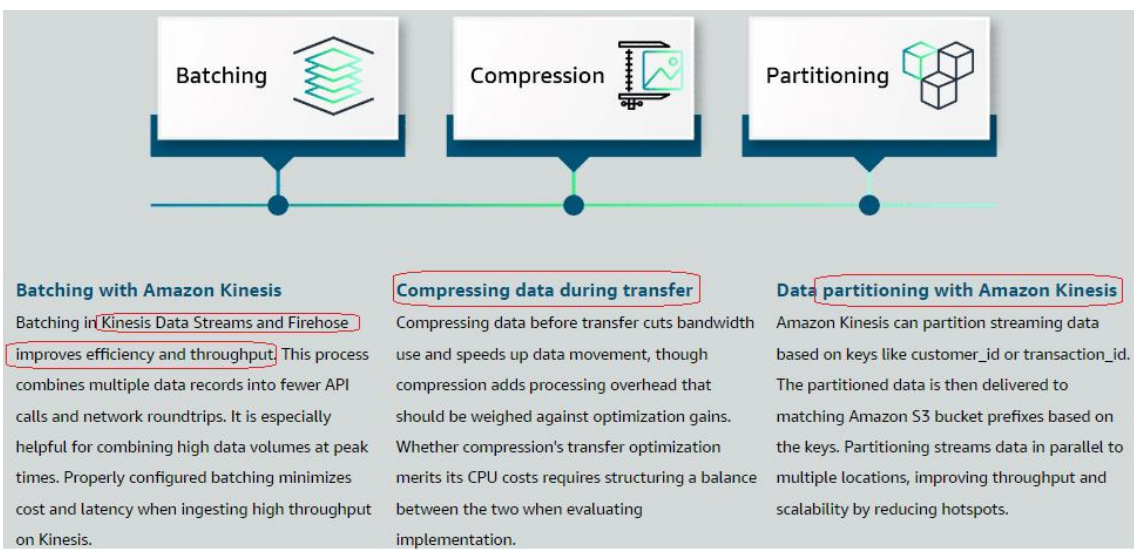
Consider optimizing network configurations or use AWS services with improved performance capabilities, such as provisioned IOPS volumes for EBS. Using techniques, such as compression or batching, can also lead to improved data-transfer efficiency.

- ***Uneven distribution of data access***

Hotspots or bottlenecks in storage systems can be caused by uneven distribution of data access resulting in performance degradation or data availability issues.

AWS services, such as S3 and EFS, automatically distribute data and provide load balancing capabilities. Data partitioning is another strategy that can be implemented, which distributes data across multiple storage resources, reducing the likelihood of hotspots.

Ingestion modifications



DO THE ASSESSMENT!!

1.1.4 Summary

Keywords/Concepts	AWS Service/Option
• Data lake, central storage, scalable, durable • Large static datasets, archival	Amazon S3
High-performance storage, real-time predictions, pre-trained models	Amazon EBS
• Concurrent access, shared datasets, scalable file system • Real-time/streaming workloads	Amazon EFS
High-performance computing, distributed architecture, Lustre	Amazon FSx
Training workloads, high IOPS, random access	• EBS (especially io2) • Instance Store
Inference workloads (standard)	• EBS gp3 • EFS
Inference workloads (low-latency)	EBS io2
Copy and load pattern Sequential streaming pattern	S3 to EBS
Randomized access pattern	FSx or EFS
Columnar storage, data compression	Parquet, ORC
Row-based storage, sequential records	CSV, Avro RecordIO
Hierarchical data, flexible structure	JSON, JSONL

Data Types and Processing

Data Type	Suitable For	Not Suitable For
Tabular data	Linear regression, classification	Complex pattern recognition in unstructured data
Columnar data	Data analysis, efficient querying	Frequent record updates
Time series data	Trend analysis, forecasting	Non-temporal pattern recognition
Image data	Object recognition, image classification	Text-based analysis
Text data	Natural language processing, sentiment analysis	Image or numerical analysis

Data Formats

Format	Suitable For	Storage Type	Characteristics
CSV	*Simple tabular data, easy human readability*	*Row-based*	*space-efficient*
Avro RecordIO	*ML workloads requiring multiple dataset iterations*		*schema-defined*
Parquet	*Large-scale data analysis*	*columnar*	*Efficient compression, fast querying*
ORC	*Big data analysis (Hive, Spark)*		*Optimized for large-scale data processing*
JSON/JSONL	*Hierarchical, non-tabular data*	*Hierarchical*	*Flexible structure, easy parsing*

Data Visualization

See Above

AWS Storage Options

Service	Advantages	Limitations
Amazon S3	*Scalable, durable, central data lake*	*Higher latency compared to local storage*
Amazon EBS	*High IOPS, suited for databases and ML training*	*Limited to single EC2 instance, requires volume management*
Amazon EFS	*Concurrent access, scalable shared file system*	*Higher costs, performance dependent on network speed*
Amazon FSx	*Lowest latency, high-performance computing*	*Potential vendor lock-in, complex management for some file system types*

Data Access Patterns

Pattern	Best With	Characteristics
Copy and Load	*S3 to EBS*	*Data copied entirely before processing*
Sequential Streaming	*S3 to EBS*	*Data streamed in batches or individual records*
Randomized Access	*EFS, FSx*	*Random data access, shared file system*

Use Case Recommendations

Use Case	Recommended Service	Reason
Training Workloads	• *EBS (io2)* • *Instance Store*	*High IOPS, low-latency random access*
Inference Workloads (standard)	• *EBS (gp3)* • *EFS*	*Balance of performance and cost*
Inference Workloads (low-latency)	*EBS (io2)*	*Higher IOPS for faster response times*
Real-time/Streaming Workloads	*EFS*	*Concurrent access, shared datasets*
Large Static Datasets	*S3*	*Cost-effective for infrequently accessed data*
Distributed Processing	• *EFS* • *FSx*	*Concurrent access from multiple instances*

1.2 Transform Data (Data Cleaning, Categorical encoding, Feature Engineering)

Remember

- ***Data cleaning*** *focuses on handling issues like missing data and outliers.*
- ***Categorical encoding*** *- Used to convert values into numeric representations.*
- ***Feature engineering*** *focuses on modifying or creating new features from the data, rather than encoding features.*

1.2.1 Data Cleaning

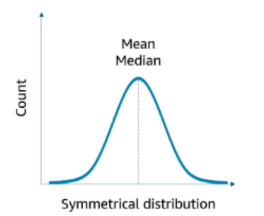
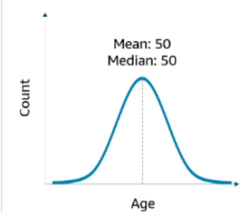
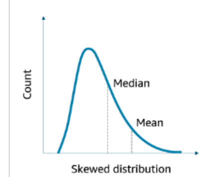
Incorrect and Duplicate Data

deduplication *- The process of automating data duplication removal. Deduplication works by scanning datasets for duplicate information, retaining one copy of the information, and replacing the other instances with pointers that direct back to the stored copy. Deduplication can drastically increase storage capacity by keeping only unique information in your data repositories.*

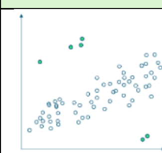
Data Outliers

Methods

1. Calculating mean and median

Mean	Median
The mean is the average of all the values in the dataset. Mean can be a useful method for understanding your data when the data is symmetrical. For example, a symmetrical set of data that contains ages of respondents might reflect that both the mean and median of the dataset is 50 years old.	The median is the value in the dataset that divides the values into two equal halves. If your data is skewed or contains outliers, the median tends to provide the better metric for understanding your data as it relates to central tendency. For example, a dataset that contains income level might contain outliers. The mean might skew toward higher or lower values, while the median would provide a more accurate picture of the data's central tendency.
 	 

2. Identifying natural and artificial outliers

Natural outliers	Artificial outliers	
Natural outliers are data points that are accurate representations of data, but are extreme variations of the central data points. For example, in a dataset that includes height measurements of individuals, an extremely tall individual would represent a natural outlier.	Artificial outliers are anomalous data points in your dataset due to error or improper data collection. For example, a faulty sensor in a thermometer might produce a body temperature that is unrealistically high or low compared to expected body temperatures.	 Scatterplot graph with outliers.

Participant ID	Age	Income	Education	State	Flu shot	Outbreak zone?
123456	39	45,000/year	Four-year degree	NY	Yes	No
123457	23	1,500/month	Baccalauréat	MN	No	Yes
123458	78		Masters/PhD		Yes	Yes
123459	20	3,000,000/year	High-school diploma	CA	No	No
123450	154	53,000/year		Masters/PhD		
123456	39	45,000/year	Four-year degree	NY	Yes	No

1 ***Artificial outlier*** *- This data is in the correct format for the Age column, but an entry of 154 is unrealistic. In this case, it makes the most sense to delete this entry from your data.*

2 ***Natural outlier*** *- Although three million dollars a year is drastically more than the rest of the salaries in our dataset, this number is still plausible. In this case, you can choose to transform the outlier and reduce the outlier's influence on the overall dataset. You will learn about that type of data transformation later in this course.*

Incomplete and Missing Data

There are some key steps you can take to address incomplete and missing values in your dataset.

a) Identify missing values

There are certain Python libraries, such as Pandas, that you can use to check for missing values.

b) Determine why values are missing

Before you can determine how to treat the missing values, it's important to investigate which mechanisms caused the missing values. The following are three common types of missing data:

- ***Missing at Random (MAR):*** *The probability that a data point is missing depends only on the observed data, not the missing data.*

  Example*: In a dataset of student test scores, scores are missing for some students who were absent that day. Absence is related to performance.*

- ***Missing Completely at Random (MCAR):*** *The probability that a data point is missing does not depend on the observed or unobserved data.*

  Example*: In an employee survey, some people forgot to answer the question about their number of siblings. Their missing sibling data does not depend on any values.*

- ***Missing Not at Random (MNAR):*** *The probability that a data point is missing depends on the missing data itself.*

  Example: In a financial audit, companies with accounting irregularities are less likely to provide complete records. The missing data depends on the sensitive information being withheld.

- ***Drop missing values***

Depending on what is causing your missing values, you will decide to either drop the missing values or impute data into your dataset.

One of the most straightforward ways to deal with missing values is to remove the rows of data with missing values. You can accomplish this by using a Pandas function. Dropping rows or columns will make the dataset non-missing. However, the risk of dropping rows and columns is significant.

Issues

- *If you have hundreds of rows or columns of data, all of that missing data might cause bias in your model predictions.*
- *If you drop too much data, you might not have enough features to feed the model.*

- ***Impute values***

Missing values might be related to new features that haven't included your dataset yet. After you include more data, those missing values might be highly correlated with the new feature. In this case, you would deal with missing values by adding more new features to the dataset. If you determine the values are missing at random, data imputation, or inputting the data into your dataset, is most likely the best option.

One common way to impute missing values is to replace the value with the mean, median, or most frequent value. You would select the mean, median, or most frequent value for categorical variables. You would select the mean or median for numerical variables. Choosing the mean, median, or most frequent value depending on your business problem and data collection procedures.

1.2.2 Categorical encoding

Categorical encoding is the process of manipulating text-based variables into number-based variables.

When to encode

Not all categorical variables need to be encoded. Depending on your use case, different ML algorithms might not require you to encode your variables.

For instance, a random forest model can handle categorical features directly. You would not need to encode values, such as teal, green, and blue, as numeric values.

Encoding Types (or types of Categorical values)

- ***Binary** categorical values refer to values that are one of two options. For example, a column of data might be true or false, such as if someone attended an event.*
- ***Nominal**, or multi-categorical, values are category values where order does not matter. A set of data that contains different geographic locations, such as states or cities, might be considered multi-categorical.*
- ***Ordinal**, or ordered, values are category values where the order does matter, like what size of drink you order at a coffee shop: small, medium, or large.*

Encode Techniques

Not all categorical variables

- ***Label encoding** converts categorical values into binary numbers*

Category value	Encoded value
Teal	0
Blue	1
Green	2

- ***One Hot encoding** : creating a new binary feature for each unique category value. Rather than assigning a different value for each data point like binary encoding, one-hot encoding sets the feature to 1 or 0 depending on if the category that applies to a given data point.*

Category value	Teal	Blue	Green
Teal	1	0	0
Blue	0	1	0
Green	0	0	1

***When to use which:** **Label coding** might not be the best technique if there are a lot of categories. In One hot encoding, these additional columns might grow your dataset so much that it makes it difficult to analyze efficiently.*

1.2.3 Feature Engineering

Feature engineering is a method for transforming raw data into more informative features that help models better capture underlying relationships in the data.

Feature Engineering by data type (numeric, text, image, and sound data types)

We only cover numeric and text here

***Numeric feature engineering** involves transforming numeric values for the model and is often accomplished by grouping different numeric values together.*

***Text feature** engineering involves transforming text for the model and is often accomplished by splitting the text into smaller pieces.*

1. *Numerical Feature Engineering*

- ***Purpose:*** *Aims to transform the numeric values so that all values are on the same scale.*
- ***Why:*** *This method helps you to take large numbers and scale them down, so that the ML algorithm can achieve quicker computations and avoid skewed results.*

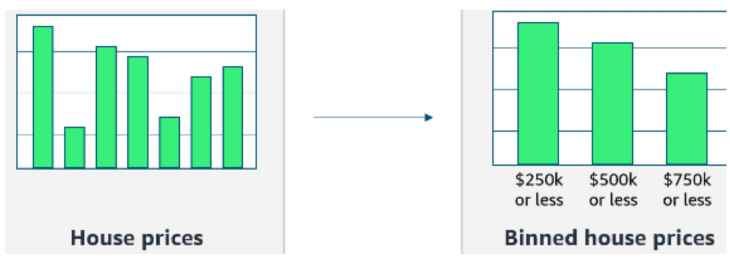
a) ***Feature Scaling:***

<table>
<tr><th>Normalization</th><th>Standardization</th></tr>
<tr><td>rescales the values (often between 0 and 1)</td><td>Similar, but mean of 0 and standard deviation of 1</td></tr>
<tr><td></td><td>When to use: Reduces the negative effect of outliers.</td></tr>
<tr><td>
<table>
<tr><th>Price (in dollars)</th><th>price_scaled</th></tr>
<tr><td>100,000</td><td>0.1</td></tr>
<tr><td>437,000</td><td>0.437</td></tr>
<tr><td>1,000,000</td><td>1</td></tr>
</table>
</td><td>
<table>
<tr><th>Price (in dollars)</th><th>price_scaled</th></tr>
<tr><td>100,000</td><td>-1.15</td></tr>
<tr><td>437,000</td><td>0</td></tr>
<tr><td>1,000,000</td><td>1.15</td></tr>
</table>
</td></tr>
</table>

b) ***Binning:***

The data is divided into these bins based on value ranges, thus transforming a numeric feature into a categorical one.

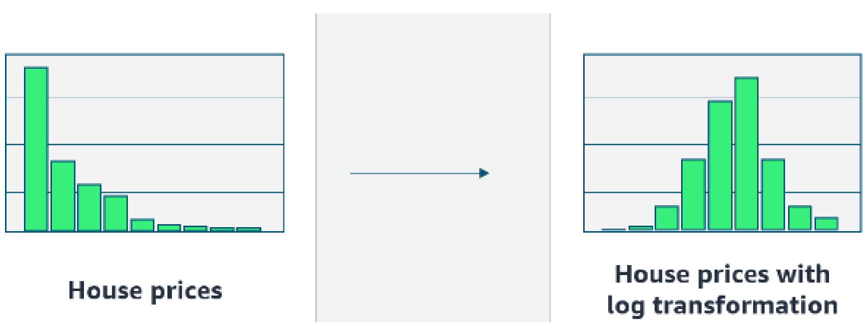
When: *numerical data when the exact difference in numbers is not important, but the general range is a factor.*



c) ***Log transformation:***

The most common logarithmic functions have bases of 10 or e, where e is approximately equal to 2.71828. Logarithmic functions are the inverse of exponential functions and are useful for modeling phenomena where growth decreases over time, like population growth or decay.

When: *When skewed numeric data, or multiple outliers. Basically, log compresses data to use lower numbers*



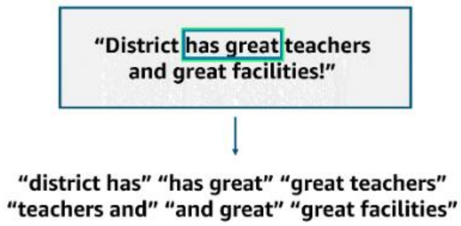
For example, the log of $10,000 would be around 4 and the log of $10,000,000 would be around 7. Using this method, the outliers are brought much closer to the normal values in the remainder of the dataset.

2. ***Text Engineering***

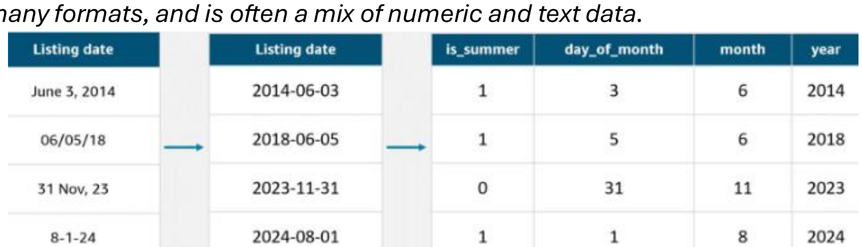
a) ***Bag of Words****: The bag-of-words model does not keep track of the sequence of words, but counts the number of words in each observation. Bag-of-words uses tokenization to create a statistical representation of the text.*

ID	Word	Count
1	District	1
2	has	1
3	great	2
4	teachers	1
5	and	1
6	facilities	1

b) ***N-gram: builds off of bag-of-words by producing a group of words of n size.***



c) ***Temporal data:*** *Temporal or time series data is data involving time, like a series of dates. Temporal data can come in many formats, and is often a mix of numeric and text data.*



Listing date
June 3, 2014
06/05/18
31 Nov, 23
8-1-24

Listing date
2014-06-03
2018-06-05
2023-11-31
2024-08-01

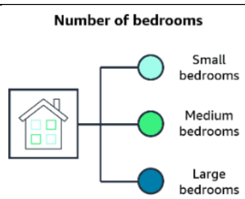
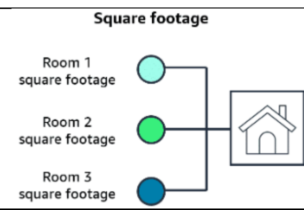
is_summer	day_of_month	month	year
1	3	6	2014
1	5	6	2018
0	31	11	2023
1	1	8	2024

When to use which

Bag of Words	*create a statistical representation of the text*
N-grams	*Phrase of n-size important (like sentiment analysis)*
Temporal data	*Capture key trends, cycles, or patterns over time*

Feature Selection Techniques

- ***Feature splitting & Feature combining***

<table>
<tr><th>Feature splitting</th><th>Feature combining</th></tr>
<tr><td>breaks down features into multiple derived features</td><td></td></tr>
<tr><td>
</td><td>
</td></tr>
</table>

- ***Principal component analysis***

  Statistical technique that you can use for dimensionality reduction

Property number	1 Square footage	Bedrooms	Bathrooms	2 Price (in dollars)	Tax rate	State
1	1,500	3	2	250,000	0.82%	NC
2	2,000	4	3	350,000	0.87%	WA
3	1,000	2	1	150,000	1.61%	WI
4	3,000	5	4	450,000	2.08%	IL
5	2,500	4	3	400,000	1.63%	NE

1 ***Principal component – Size{|:*** *The first component accounts for the physical attributes of the home that includes square footage, bedrooms, and bathrooms*

2 ***Principal component - Cost:*** *The second component captures the financial factors that include price and tax rate.*

X. AWS Tools for Data Transformation

X.1. Data Labeling with AWS

1. Mechanical Turk

Purpose:

- ***Image annotation:***
- ***Text annotation:***
- ***Data collection:***
- ***Data cleanup:***
- ***Transcription:***

2. SageMaker Ground Truth

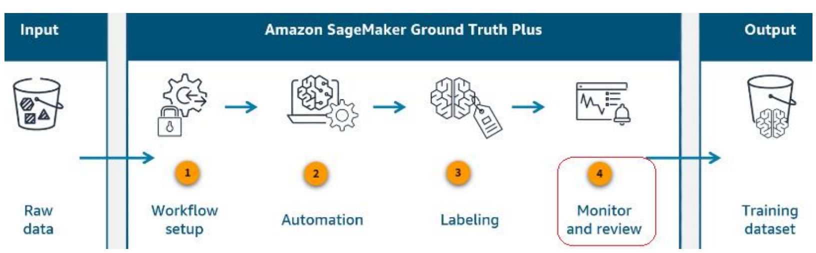
Uses Mechanical Turk and other data processing methods to streamline the data preparation process even further.

Purpose

- ***Image annotation:***
- ***Text annotation:***
- ***Object Detection***
- ***Named Entity Recognition***

3. SageMaker Ground Truth Plus

Fully managed data labeling service using expert labelers.



When to use which

Mechanical Turk *On-demand access to workers, lower costs, fast turnaround times, task flexibility, and quality control capabilities.*
SageMaker Ground Truth *Higher quality, or Object detection or NER, public sourcing*
SageMaker Ground Truth Plus *Production labeling workflows, sensitive data, complex tasks, and custom interfaces*

X.2. Data Ingestion with AWS

1. ***Data Wrangler***

   ***Purpose**: visual, code-free tool for data preprocessing and feature engineering*

   Steps

   - ***Clean data:***
   - ***Feature engineering:** Combine columns and apply formulas, etc.*
   - ***Fixing formatting issues:** Missing headers, encoding problems, etc.*
   - ***Reducing data size:** For large datasets*
   - ***Automating transformations:***

2. ***SageMaker Feature Store***

   ***What**:*

   - *Managed repository for storing, sharing, and managing features.*
   - ***Storing features** saves time by eliminating redundant feature engineering efforts.*

   Steps

   - *Automated data preprocessing*
   - *Centralized feature repository:*
   - *Management of feature pipelines:*
   - *Standardized features:*
   - *Caching for performance:*

When to use which

- ***Use Data Wrangler:***
  - *Initial data exploration*
  - *one-time transformations*
  - *when working directly in notebooks.*
- ***Use Feature Store***
  - *Moving to production*
  - *Sharing features across models or teams*
  - *when you need low-latency feature serving for online inference.*

X.3. Data Transformation with AWS

1. Data Glue

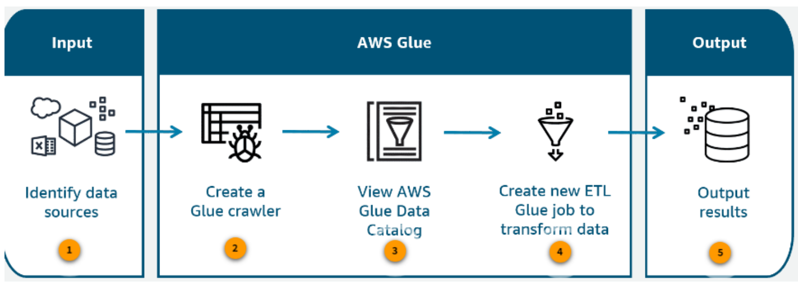
Purpose:

- *AWS Glue auto-generates Python code to handle issues like distributed processing, scheduling, and integration with data sources.*
- *AWS Glue DataBrew is visual data preparation tool for cleaning, shaping, and normalizing datasets.*

Use cases

- *Automated ETL pipelines*
- *Data integration and ingestion:*
- *Data cleansing and standardization*
- *Feature engineering:*
- *Final pretraining data preparation*

Steps



2. SageMaker Data Wrangler

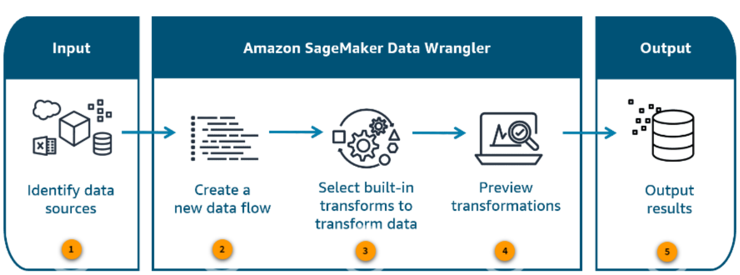
Purpose:

- *We know SageMaker Data Wrangler to ingest data.*
- *SageMaker Data Wrangler can also help explore, clean, and preprocess data without writing code.*

Use Cases

- *Clean Data & Fix formatting issues*
- *Feature Engineering*
- *Reducing data size*
- *Automating transformations*

Steps



When to use:

- ***Glue**: Production ETL pipelines, large-scale data processing, scheduled jobs*
- ***Data Wrangler:** Exploratory data analysis, quick transformations, ML data prep in SageMaker*

3. For Streaming data

Lambda	***Spark on Amazon EMR***
• *Data normalization:* • *Data filtering:* • *Transcoding media:*	• *Real-time analytics:* • *Anomaly detection:* • *Monitoring dashboards*

1.3 Validate Data and Prepare for Modeling

1.3.1 VALIDATE DATA

Basics

Bias Metrics

	Class imbalance (CI)	Difference in proportion of labels (DPL)
	Occurs when distribution of classes in the training data is skewed (one class significantly less represented than the others).	*Compare the distribution of labels in data*
Understand	*If CI +ve, advantaged **group** is relatively overrepresented in this dataset.* *If CI -ve, advantaged **group** is relatively underrepresented in this dataset.*	*If DPL +ve, one **class** significantly higher proportion.* *If DPL -ve, one **class** significantly less proportion.*

Data Validation Strategies

	Resampling	*Synthetic data generation*	*Data augmentation*
Add/Update	Adding data	Adding Data	Transform data
Auto/Manual	Manually	Algorithmically	Algorithmically
How	**Oversample/under sample**	**Creating new artificial data points**	
Technique(s)		SMOTE	GAN
Type of Data	Numeric	Text based	Image

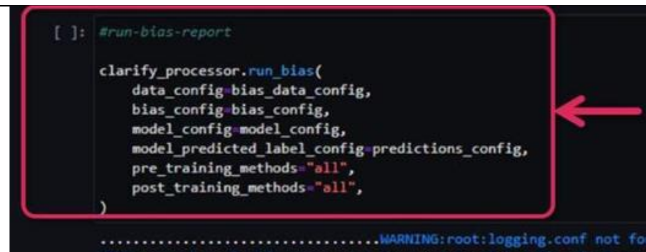
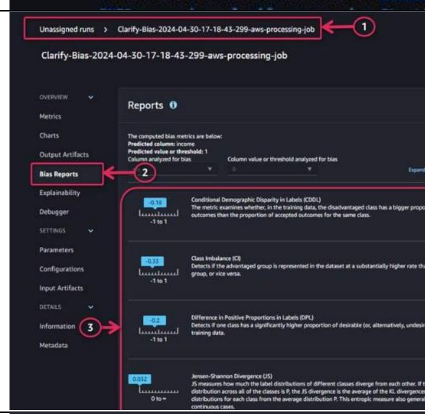
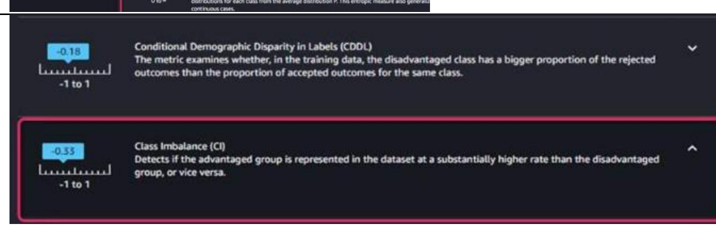
AWS tools for Data Validation

	Glue Data Quality	*Glue DataBrew*	*Comprehend*
What	Managed monitoring	Visual data prep tool	NLP tool
Use cases	• Data Validation • Data quality • Automated scheduling • Data quality dashboards	• Data profiling • Built-in transformations • Custom transformations	• Entity recognition • Language detection • Topic modeling

SageMaker Clarify

How it works

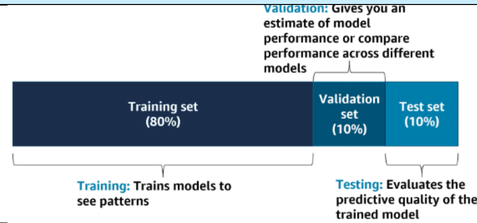
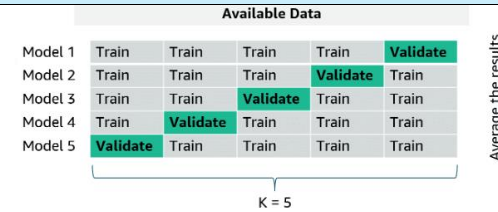
- *Create a bias report using the configuration for pre-training and post-training analysis*
- *Assess the bias report by considering the class imbalance (CI) and difference in proportion of labels (DPL).*

<table>
<tr><td>1. Set up the bias report: To set up your bias report configurations, use the BiasConfig to provide information on which columns contain the facets with the sensitive group of sex, what the sensitive features might be using facets_values_or_threshold, and what the desirable outcomes are using labels_values_or_threshold.</td><td></td></tr>
<tr><td>2. Run the bias report: create the bias report using the configuration for pretraining and post-training analysis. This step takes approximately 15-20 minutes</td><td>
</td></tr>
<tr><td>3. Access the bias report:</td><td>
</td></tr>
<tr><td>4. Assess the Class Imbalance: The CI shows -0.33 as the bias metric. CI measures if the advantaged group, men, is represented in the dataset at a substantially higher rate than the disadvantaged group, women, or the other way around. The negative value demonstrated indicates that the advantaged group, men, is relatively underrepresented in this dataset example.</td><td>
</td></tr>
<tr><td>5. Assess the Difference in Positive Proportion of Labels:
Review the DPL. This metric detects a label imbalance between classes that might cause unwarranted biases during training.</td><td></td></tr>
</table>

1.3.2 PREPARE FOR MODELLING

Dataset Splitting, Shuffling, and Augmentation

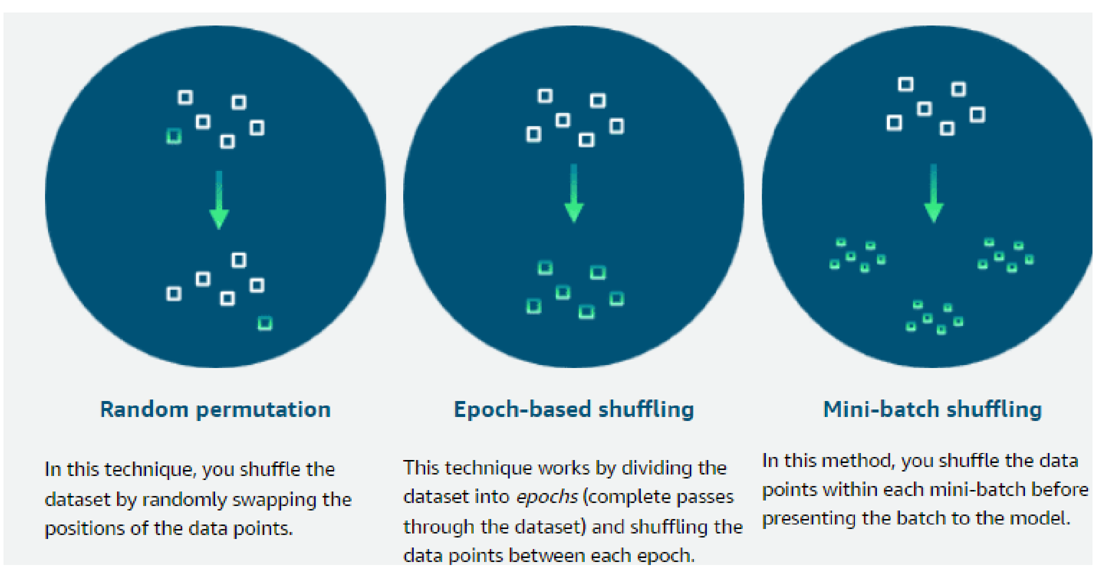
Data Splitting Techniques

	Train, test, validate	***Cross-validation***
		
Best for	• Easy implementation • Provides quick estimate of model performance	• Uses **the entire dataset** for training and testing, maximizing data usage • Reduces variance in performance estimation **by averaging results across multiple iterations**
Limitations	• Performance estimate might have variance due to dependency on specific examples in the test set • Not suitable for small datasets because it might lead to overfitting or underfitting	• Computationally more expensive, especially for large datasets • Might be sensitive to class imbalances if not stratified properly
Example	•	**K-fold cross-validation**

Dataset shuffling

Benefits of data shuffling: *Dataset shuffling plays a crucial role in mitigating biases that might arise from the inherent structure of the data. By introducing randomness through shuffling, you can help the model be exposed to a diverse range of examples during training.*

Data shuffling techniques:



Dataset Augmentation

Data augmentation works by creating new, realistic training examples that expand the model's understanding of the data distribution. Dataset augmentation involves artificially expanding the size and diversity of a dataset

Data augmentation techniques:

- *Image-based Augmentation*
    - *Flipping, rotating, scaling, or shearing images*
    - *Adding noise or applying color jittering*
    - *Mixing or blending images to create new, synthetic examples*
- *Text-based Augmentation*
    - *Replacing words with synonyms or antonyms*
    - *Randomly inserting, deleting, or swapping words*
    - *Paraphrasing or translating text to different languages*
    - *Using pre-trained language models to generate new, contextually relevant text*
- *Time series Augmentation*
    - *Warping or scaling the time axis*
    - *Introducing noise or jitter to the signal*
    - *Mixing or concatenating different time-series segments*
    - *Using generative models to synthesize new time-series data*

When to use which

- ✓ ***Data-splitting (Train, Test, Validate):***
    - ***Pros:*** *Clear separation of data, prevents data leakage*
    - ***Cons:*** *Reduces amount of data available for training*
- ✓ ***Cross-validation:***
    - ***Pros:*** *Makes efficient use of all data, robust performance estimate*
    - ***Cons:*** *Computationally expensive, may not be suitable for very large datasets*
- ✓ ***Data shuffling:***
    - ***Pros:*** *Reduces bias, improves generalization*
    - ***Cons:*** *May not be appropriate for time-series data where order matters*
- ✓ ***Data augmentation:***
    - ***Pros:*** *Increases dataset size, improves model robustness*
    - ***Cons:*** *May introduce artificial patterns if not done carefully*

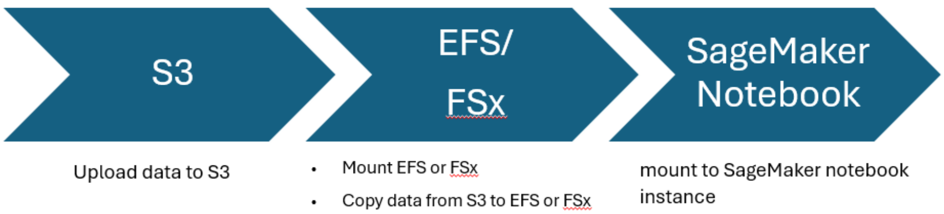
AWS services for pre-training data configuration

Final formatting process

SageMaker built-in algorithms for formatting

- **CSV**: Many built-in algorithms in SageMaker:
  - XGBoost
  - linear learner
  - DeepAR.
- **RecordIO-protobuf:** Commonly used for image data, where each record represents an image and its associated metadata.

Steps after formatting:



- **Upload data to Amazon S3**
- **Mount Amazon EFS or Amazon FSx**
- **Copy data from Amazon S3 to EFS** or **FSx**
  - Use AWS data transfer utilities or custom script
  - Verify data integrity by checking file sizes and checksums after the transfer is complete.
- **Load the data into your ML training resource**
  - With Amazon EFS -> you would create an EFS file system and mount it to your SageMaker notebook instance or training job. Copy dataset files into the EFS file system. Then in your training script, load the data by accessing the Amazon EFS mount path.
  - For Amazon FSx -> create a Lustre file system and attach it to your SageMaker resource. Copy the data files to the FSx Lustre file system. In your training script, load the data by accessing the Amazon FSx mount path.
  - Note that both Amazon EFS and Amazon FSx for Lustre provide shared file storage that can be accessed from multiple Amazon Elastic Compute Cloud (EC2) instances at the same time.
- **Monitor, refine, scale, automate, and secure**
  - When your data is loaded into your resource, you will continue to monitor, refine, scale, automate, and secure your ML workloads. Monitoring, refining, scaling, automating, and securing your workloads is a complex and involved part of the ML lifecycle.
  - Implement data lifecycle management strategies, such as archiving or deleting old or unused data.
  - Consider using AWS services like AWS Step Functions, AWS Lambda, Amazon Managed Workflows for Apache Airflow (Amazon MWAA), and AWS CodePipeline to automate and orchestrate your data workflows.