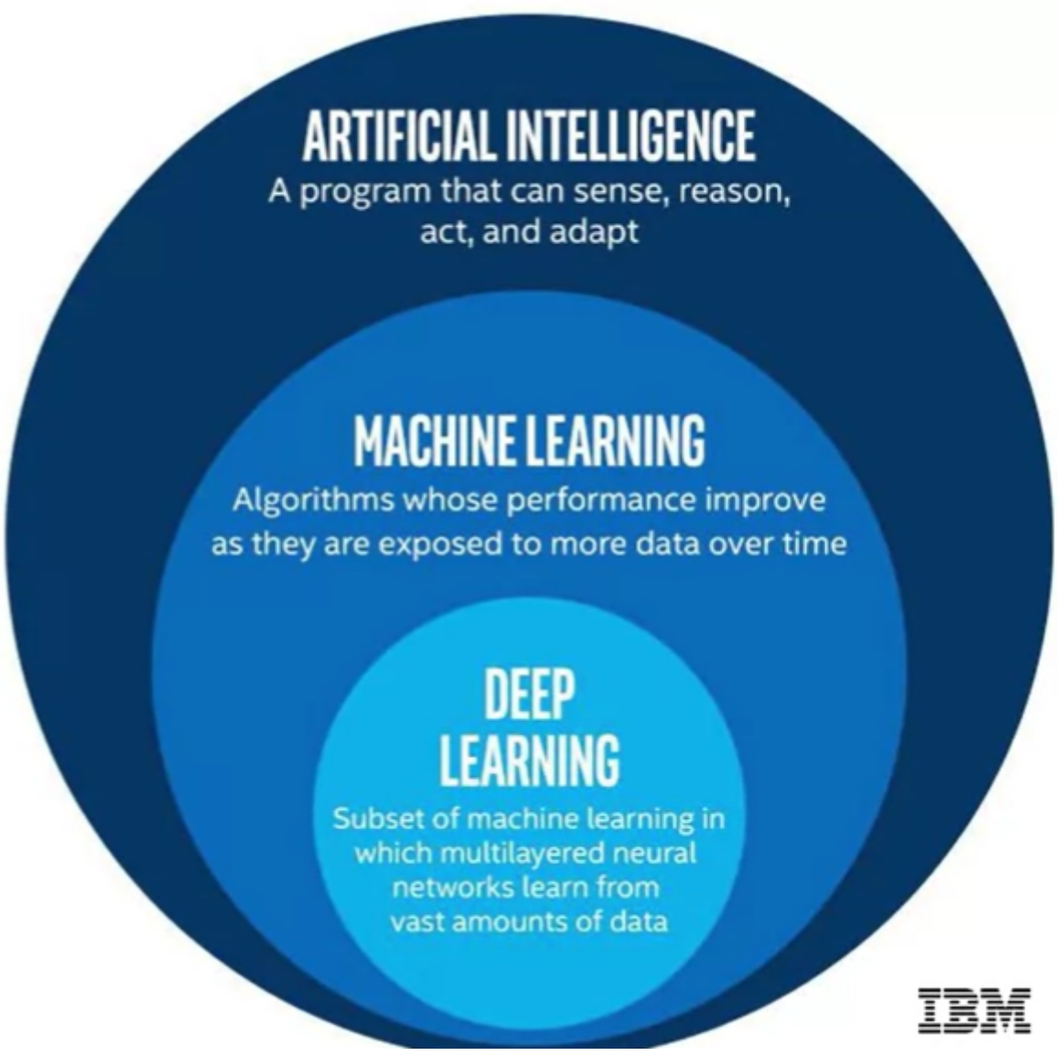
Course I - Exploratory Data Analysis for ML

1.1 Introduction to AI, ML, and Deep Learning

Definition and Relationship:
- AI encompasses machines exhibiting intelligent behavior.
- Machine learning (ML) is a subset of AI where machines learn from data.
- Deep learning (DL) is a subset of ML utilizing multi-layered neural networks.



Advancements and Impact:
- DL has overcome limitations of classical ML, propelling AI into a new era.
- History of AI includes ups and downs, leading to today's AI boom.
- Current AI advancements differ from past advancements.
- Real-world applications include image processing, machine translation, advertising, supply chain optimization, self-driving cars, and smart homes.

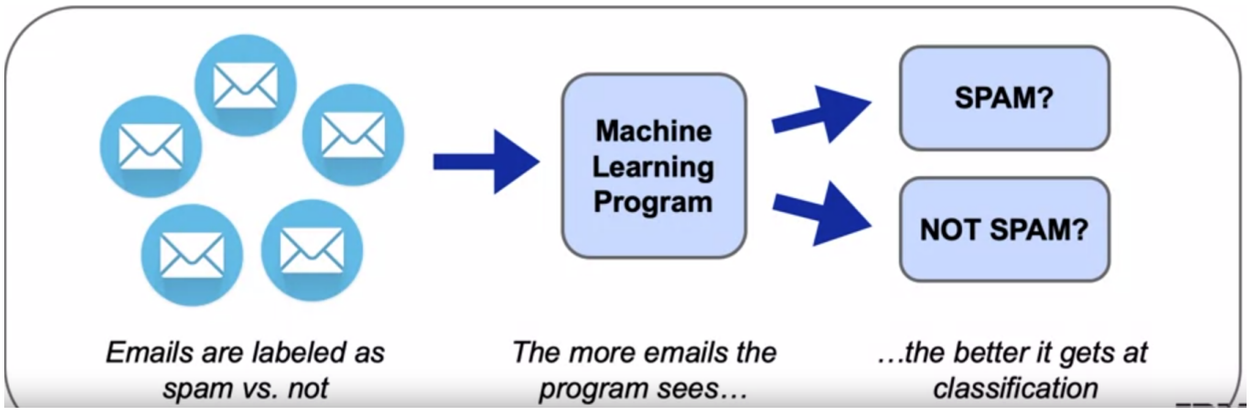
Breakthroughs and Future Impact:
- Image classification and machine translation as major breakthroughs.
- DL and innovations in data storage and processing power enable progress.
- AI's potential impact across industries comparable to electricity's impact a century ago.

Definitions and Perspectives:
- AI defined as simulating intelligent behavior in computers.
- AI colloquially associated with mimicking human cognitive functions.
- Examples of AI beyond ML and DL include rule-based systems.

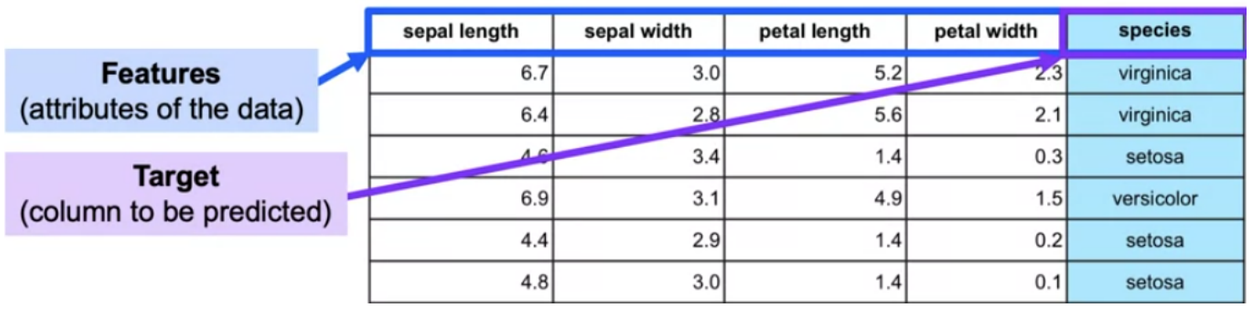
Definition and Concept:
- Machine Learning (ML) involves programs learning patterns from data rather than explicit programming by humans.
- ML is a subset of Artificial Intelligence (AI) where algorithms improve with exposure to more data.
- Performance of ML algorithms may plateau with diminishing returns after a certain amount of data.
- Machine Learning (ML) involves the study and construction of programs that learn patterns from data, rather than being explicitly programmed by humans.
- ML is a subset of Artificial Intelligence (AI), focusing on learning from data to improve performance.

Learning from Data:
- ML programs learn from exposure to more data over time, improving their ability to recognize underlying patterns.
- Performance may plateau after a certain amount of data, leading to diminishing returns.

Example: Email Spam Detection:
- Dataset consists of labeled emails (spam vs. not spam).
- ML algorithm learns patterns from labeled data to distinguish between spam and non-spam emails.
- Trained algorithm can then predict whether new emails are spam or not.



Features and Target:
- Features: Characteristics used for prediction (e.g., sepal length, width, petal length, width).
- Target: Variable being predicted (e.g., iris species).

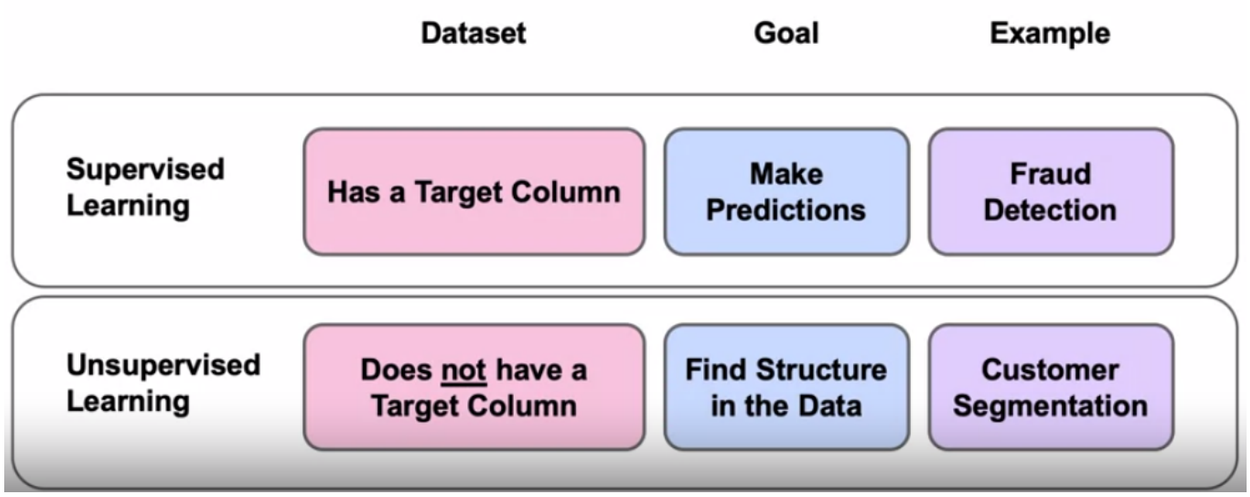


sepal length	sepal width	petal length	petal width	species
6.7	3.0	5.2	2.3	virginica
6.4	2.8	5.6	2.1	virginica
4.6	3.4	1.4	0.3	setosa
6.9	3.1	4.9	1.5	versicolor
4.4	2.9	1.4	0.2	setosa
4.8	3.0	1.4	0.1	setosa

Types of Machine Learning:

- **Supervised Learning:**
  - Dataset includes a target column or labels.
  - Goal is to predict the label based on input features.
  - Example: Email spam detection, iris flower classification.
- **Unsupervised Learning:**
  - Dataset does not include a target column.

- Goal is to find underlying structure or patterns in the data.
- Example: Customer segmentation, fraud detection.

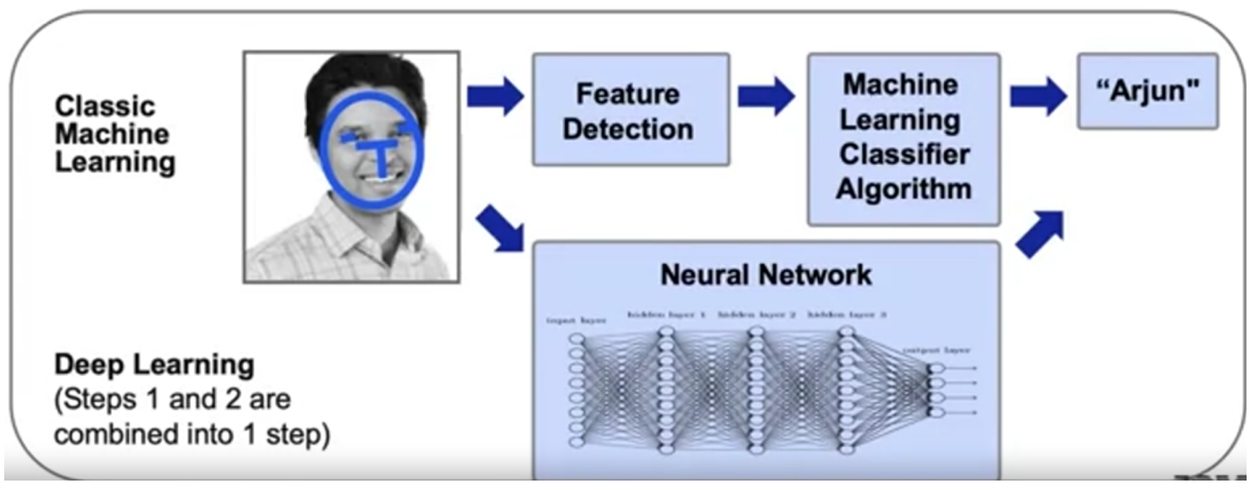


Applications:

- **Supervised Learning:**
  - Example: Fraud detection in credit card transactions.
  - Features such as transaction time, amount, location, and category are used to predict fraud.
- **Unsupervised Learning:**
  - Example: Customer segmentation in e-commerce data.
  - No predefined labels; grouping similar customers for targeted marketing.

Challenges and Limitations:
- Structured data with intuitive features is suitable for traditional ML tasks.
- Unsupervised learning may require testing different models to identify meaningful patterns.
- Traditional ML techniques perform well with structured data, while deep learning is effective for complex tasks like image classification.

Challenges in Image Feature Extraction:
- Defining features in images is complex due to the sheer volume of pixels.
- Each pixel can be considered a feature, resulting in a large number of features for even small images (e.g., 65,000 features for a 256x256 pixel image).
- Treating each pixel individually disregards spatial relationships between neighboring pixels, which are crucial for image interpretation.



Deep Learning techniques excel in learning hierarchical representations of data.

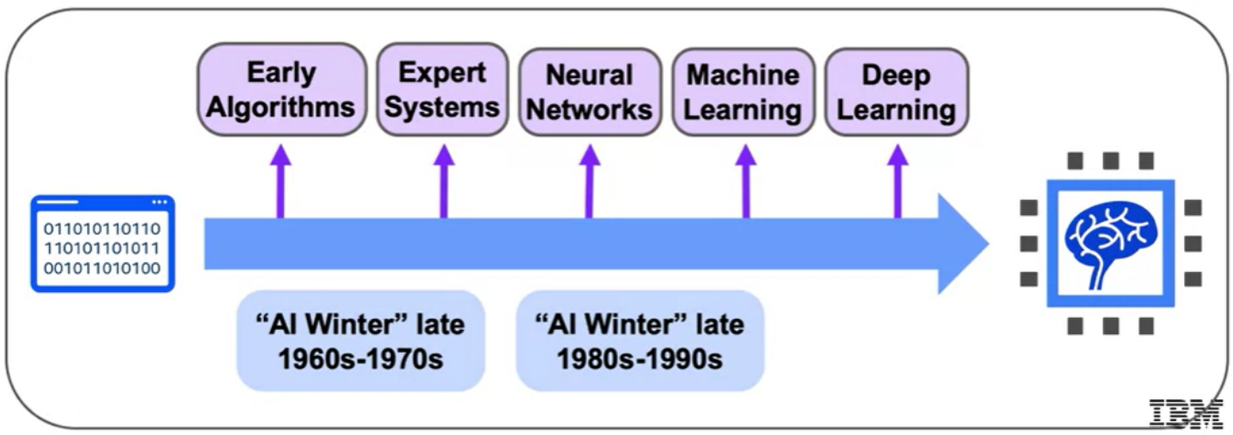
- Deep neural networks (DNNs) automatically learn features from raw data, such as images.

- DNNs can capture complex spatial relationships between pixels, which traditional Machine Learning techniques struggle with.
- By learning features from data, deep learning bypasses the need for manual feature engineering, making it highly effective for tasks like image classification.

Introduction to Deep Learning:
- Deep learning is a subset of Machine Learning that involves using complex models, primarily deep neural networks.
- Deep neural networks determine the best representation of the original data, learning intricate patterns and relationships.
- Deep learning has revolutionized image classification and other tasks by surpassing the performance of traditional ML algorithms, particularly with large datasets.
- While deep learning excels with large datasets and complex tasks, it may not always be the best choice for smaller datasets or tasks with rapidly changing data.
- Standard Machine Learning algorithms may outperform deep learning in certain scenarios, especially when interpretability or adaptability to changing data is crucial.

Comparison with Classical ML:
- Classical ML requires manual feature engineering, where data scientists define relevant features before training the model.
- Deep Learning combines feature extraction and prediction in a single step, allowing the model to learn meaningful features directly from raw data.
- Intermediate layers in deep neural networks may contain abstract features that are not easily interpretable but are highly effective for tasks like image classification.

1.2 History of Artificial Intelligence



- **Initial Excitement (1950s):**
  - Coined the term "Artificial Intelligence" in the 1950s, sparking initial excitement and investment.
  - Development of the perceptron algorithm by Frank Rosenblatt in 1957 showcased early learning from data.
  - Arthur Samuel's checkers program in 1959 demonstrated machine learning capabilities.
- **First AI Winter (1960s-1970s):**
  - Disappointment arose due to the failure to meet high expectations.
  - Major setbacks included limitations of the perceptron algorithm and critical reports on AI progress.
- **Second AI Boom (1980s):**
  - Rise of expert systems, rule-based algorithms aiding decision-making in businesses.

    - Geoffrey Hinton's work on the Backpropagation algorithm opened doors for multi-layer neural networks.
    - Excitement around AI surged again with possibilities of complex learning models.
- **Second AI Winter (1980s-1990s):**
    - Expert systems faced limitations in learning and scalability.
    - Backpropagation algorithm struggled with large datasets and networks.
    - Investment in AI research declined as practical applications fell short of expectations.
- **Successes in Machine Learning (Late 1990s-2000s):**
    - Machine learning, particularly in areas like speech recognition and search algorithms, showed promise.
    - Breakthroughs in deep learning addressed historical limitations of neural networks, leading to significant advancements in tasks like image classification and machine translation.
- **Current State and Future Prospects:**
    - Deep learning has overcome previous barriers and outperformed classical machine learning techniques.
    - Excitement surrounds the potential of AI in various sectors, indicating a shift from previous cycles of investment and disappointment.
    - Ongoing research and developments continue to shape the landscape of artificial intelligence, with implications for various industries.

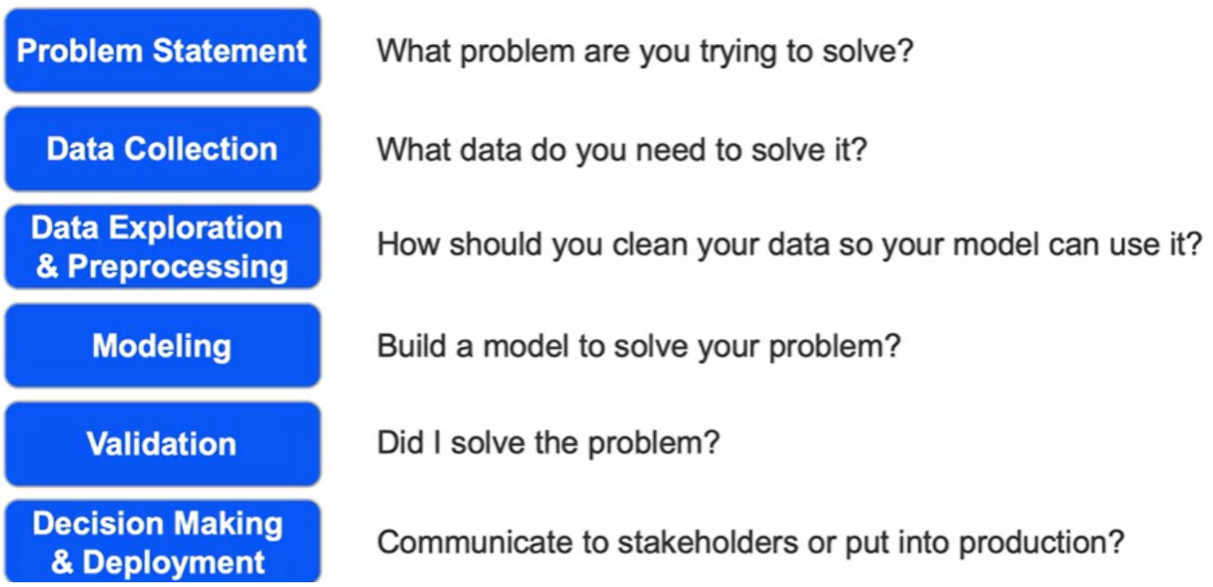
1.3 Advancements in Artificial Intelligence

- **90s and 2000s:**
    - Classical machine learning techniques saw significant successes in various areas such as speech recognition, medical diagnosis, and robotics.
    - In 1996, Deep Blue defeated a world champion chess player, marking a major milestone in AI history.
    - Google's PageRank algorithm revolutionized the search engine space, enhancing the quality of search results.
- **Overcoming Limitations of Deep Learning (2006):**
    - Algorithmic advancements addressed previous limitations of deep learning, enabling training of deeper networks.
    - Geoffrey Hinton's work on unsupervised pre-training contributed to overcoming issues like exploding and vanishing gradients.
- **Introduction of ImageNet Database (2009):**
    - ImageNet provided millions of labeled images, allowing practitioners to train models on a large scale.
    - The first ImageNet competition in 2010 further accelerated advancements in visual recognition tasks.
- **Climax: AlexNet (2012):**
    - AlexNet, a deep learning model using convolutional neural networks, achieved a significant breakthrough in visual recognition tasks.
    - This marked a turning point, leading to practical applications of deep learning in various domains.
- **Subsequent Breakthroughs:**
    - In 2013, deep learning was used for giving conceptual meaning to words, leveraging copious amounts of written data.
    - Machine translation saw breakthroughs in 2014, making it a reality with the use of recurrent neural networks.

    - Stanford researchers advanced computer vision in 2014, enabling photo annotation with descriptive captions.
    - TensorFlow, one of the most popular deep learning libraries, was introduced in 2015, enhancing the accessibility and power of deep learning.
    - DeepMind's AlphaGo defeated a Go master in 2016, showcasing advancements in game-playing AI.
    - Waymo, Google's self-driving car taxi service, launched in 2018, demonstrating progress in autonomous vehicle technology.
    - IBM Project Debater engaged in a fascinating debate with a human debater in 2019, showcasing AI's ability to argue and respond intelligently.
- **Current State and Future Prospects:**
    - These milestones highlight the significant advancements in artificial intelligence, shaping various aspects of society and technology.
    - TensorFlow 2.0 and other developments continue to make deep learning more accessible and powerful.
    - Excitement surrounds the potential of AI to drive further innovations and transformations in the future.
- **Drastic Growth in Computer Vision and Natural Language Processing (NLP):**
    - Computer vision advancements impact areas like autonomous driving and healthcare diagnostics.
    - NLP innovations include translation, sentiment analysis, clustering of news articles, and more.
- **Key Factors Differentiating This Era:**
    - **Bigger Datasets:** Access to larger and more diverse datasets, facilitated by cloud infrastructure and data capture methods, enables training of models for complex patterns.
    - **Faster Computers:** Powerful hardware for data processing and storage is now more accessible, leading to faster computations.
    - **Success of Neural Networks:** Constant innovation in deep learning has led to practical results across industries.
- **Cutting-Edge Applications Across Industries:**
    - **Healthcare:** AI aids in medical imaging, drug discovery, patient care, and sensory aids for disabilities.
    - **Industrial:** Automation in factories, predictive maintenance, optimization of agricultural production, and inventory management.
    - **Finance:** Algorithmic trading, fraud detection, personal finance, risk mitigation, and research.
    - **Energy:** Location of reserves, smart grid optimization, energy conservation, and operational improvements.
    - **Government:** Defense, threat detection, citizen insights, safety and security, and smarter city development.
    - **Transportation:** Autonomous vehicles, automated trucking, aerospace optimization, and search and rescue with drones.
    - **Education:** Personalized learning, curriculum development based on student needs.
    - **Gaming:** AI enhances player experiences by creating more human-like interactions.
    - **Service Industries:** Automation of responses and issue resolution.
    - **Telco and Media:** Personalized content recommendations for consumers.
    - **Sports:** Analytics for player recruitment, ticket pricing optimization, and business operations.
- **AI's Ubiquitous Presence:**

    - AI is deeply integrated into various aspects of society and industry, impacting everyday life.
    - This era of AI is characterized by its widespread application across diverse domains, making it different from previous periods.
- **Applications of Artificial Intelligence:**
    - **Navigation Apps:** Google Maps and Waze utilize AI to optimize routes by considering factors like traffic conditions, historical data, and weather forecasts.
    - **Ride-Sharing Platforms:** Uber and Lyft use AI to dynamically adjust prices based on supply and demand in real-time.
    - **Social Media:**
        - AI is used for content personalization, suggesting connections, and targeted advertising.
        - Image recognition and sentiment analysis help ensure appropriate content delivery.
    - **Natural Language Processing (NLP):**
        - Virtual assistants like Siri and Alexa leverage NLP to understand and respond to voice commands.
    - **Object Detection:**
        - Facebook uses AI to recognize faces in photos, facilitating easy sharing with friends.
        - Object detection is crucial for self-driving cars to identify obstacles and navigate safely.
    - **Computer Vision:**
        - Deep learning enables advanced image classification, surpassing human performance in many tasks.
        - Live detection capabilities are essential for real-time applications like self-driving cars.
    - **Future Applications:**
        - AI-powered systems can detect abandoned baggage in public spaces, potentially saving lives.
        - Real-time object detection plays a critical role in implementing such safety measures.

1.4 Basic Machine Learning Vocabulary and Workflow



1. Problem Statement:

- Define the problem you aim to solve, such as image classification or prediction of a target variable.

2. Data Collection:

- Gather relevant data needed to train and test your model, ensuring it covers various scenarios and is correctly labeled.

3. Data Exploration and Preprocessing:

- Analyze and clean the data to prepare it for modeling, including handling missing values, outliers, and converting data types if necessary.
- Techniques may include visualizing data distributions, checking for correlations, and scaling features.

4. Modeling:

- Develop and train a model to solve the problem, selecting appropriate algorithms and tuning hyperparameters.
- Start with baseline models and iterate to more complex ones as needed.

5. Validation:

- Evaluate the performance of the model using validation techniques such as cross-validation or splitting data into training and testing sets.
- Assess if the model effectively solves the problem and generalizes well to unseen data.

6. Decision-Making and Deployment:

- Based on the model's performance, decide whether to deploy it in real-world scenarios or further refine it.
- Communicate results with stakeholders and consider the practical implications of implementing the model.

Machine Learning Vocabulary:

1. Target Variable: The variable or value being predicted or classified by the model.
2. Features or Explanatory Variables: The input variables used to predict the target variable.
3. Example or Observation: A single row or data point within the dataset used for training and testing the model.
4. Label: The specific value of the target variable corresponding to an example or observation.

1.5 Data Retrieval and Cleaning

Retrieving Data from Different Sources:

1. Reading CSV Files:

- CSV (Comma Separated Values) files contain rows of data separated by commas.
- Use Pandas to read CSV files easily with a few lines of code.
- Example code:

```
import pandas as pd
file_path = 'data/iris_data.csv'
data = pd.read_csv(file_path)
print(data.iloc[:5])  # Print first five rows
```

- Useful arguments:
  - `sep`: Specify the separator (e.g., comma, tab) if not comma-separated.
  - `header`: Specify which row is the header for column names.
  - `names`: Specify custom column names.
  - `na_values`: Specify values to be treated as null.

2. Reading JSON Files:

- JSON (JavaScript Object Notation) files store data in an organized, easy-to-access manner, similar to Python dictionaries.
- Use Pandas `read_json` function to read JSON files.
- Example code:

```
data = pd.read_json(file_path)
```

- Useful arguments for reading JSON files:
  - `orient`: Specify the organization of data (e.g., 'split', 'records', 'index', 'columns', 'values').
- Writing JSON files: Use `to_json` method of DataFrame to output data to a JSON file.

Understanding how to retrieve data from various sources such as CSV files and JSON files is essential for data analysis and machine learning tasks. These methods allow for easy access and manipulation of data, facilitating further analysis and modeling.

3. Working with SQL Databases:

SQL (Structured Query Language) databases are highly structured relational databases with fixed schemas. There are various types of SQL databases, including:

- Microsoft SQL Server
- Postgres
- MySQL
- AWS Redshift
- Oracle DB
- Db2 (IBM)

Python libraries like sqlite3, SQLAlchemy, Psycopg2, and ibm_db can be used to connect to different SQL databases.

Example using sqlite3:

```
import sqlite3
import pandas as pd

# Initialize path to SQLite database
path = 'data/classic_rock.db'

# Establish connection to database
con = sqlite3.connect(path)

# Write SQL query
query = "SELECT * FROM rock_songs"

# Read data into Pandas DataFrame
data = pd.read_sql(query, con)

# Close connection
con.close()
```

4. Working with NoSQL Databases:

NoSQL databases are non-relational databases that store data in various formats, often in JSON format. Examples include:

- Document databases: Each document represents one observation.
- Graph databases: Used for network analysis, maintaining relationships.
- Wide column families: Columns are grouped together based on column families.

Example using MongoDB:

```
from pymongo import MongoClient
import pandas as pd

# Establish connection to MongoDB
con = MongoClient('mongodb://username:password@host:port/database')

# Select database
db = con.database_name

# Read data from MongoDB
cursor = db.collection_name.find({})
data = pd.DataFrame(list(cursor))

# Close connection
con.close()
```

5. Working with APIs and Cloud Data Access:

- APIs: Data providers make data available via APIs (e.g., Twitter, Amazon).
- Cloud data sources: Access data stored in the cloud (e.g., UC Irvine Machine Learning Library).

Example accessing data from an API:

```
import pandas as pd

# Define URL
data_url = 'https://example.com/data.csv'

# Read data from URL
df = pd.read_csv(data_url)
```

Data cleaning is crucial for machine learning because "garbage-in, garbage-out" applies. Messy data leads to unreliable outcomes, as models can only be as good as the data they are trained on.

Issues with Messy Data:

1. Lack of data: Insufficient relevant data can hinder model success. Companies need to collect appropriate data or acquire additional data from third parties.
2. Too much data: Having too much data across different environments and databases becomes a data engineering problem. Companies need to organize and make data ready for machine learning.
3. Bad data: Data quality management challenges arise, affecting AI adoption. Business leaders prioritize improving data use, but managing data quality remains a challenge.

Sources of Messy Data:

1. **Data duplicates:** Duplicate observations can add extra weight to models or introduce unnecessary noise.
2. **Inconsistent text and typos:** Correct spelling and formatting are crucial for consistent data categorization.
3. **Missing data:** Some degree of missing data is inevitable, but too much in the wrong fields can hinder feature usability.
4. **Outliers:** Outliers skew features disproportionately, making it difficult to find the true underlying model.
5. **Data sourcing issues:** Trouble arises when bringing in data from multiple systems or combining data from different sources, leading to mismatches.

Handling Duplicated Data:

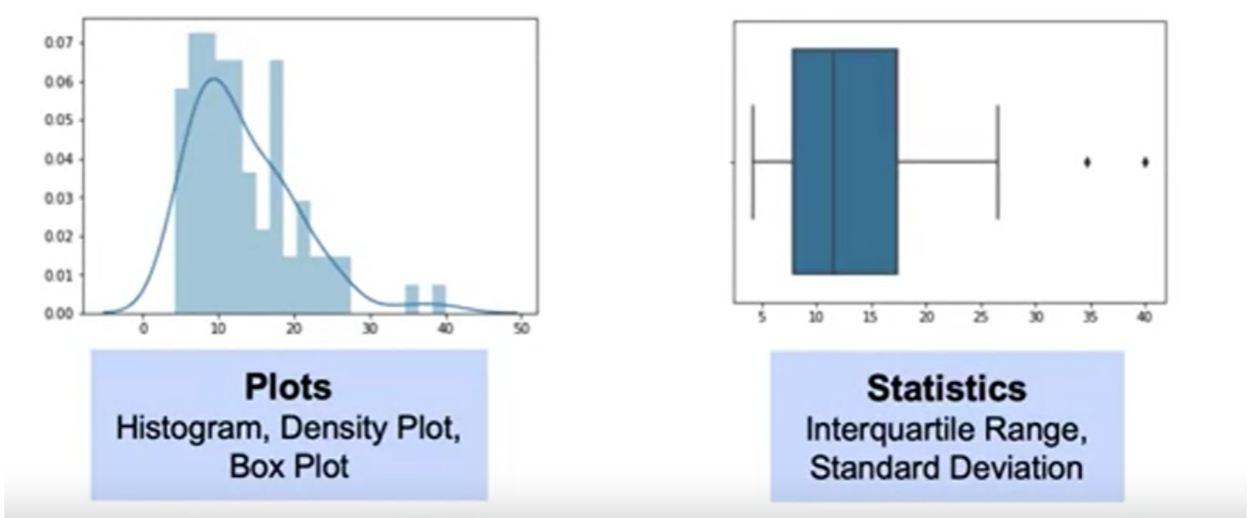
- Determine if duplicate observations are needed. For example, duplicates may be valid in datasets like Iris if two flowers are exactly the same.
- Evaluate if duplicate images are necessary for labeling tasks. Duplicate exact pictures are usually not helpful for labeling.
- Filter the data to identify duplicates but avoid filtering too much to retain access to potentially useful initial data.

Dealing with Missing Data:

When handling missing data, it's essential to ensure that every feature and label in the dataset has some information. Here are some options:

1. **Remove data:** Entire rows containing missing values can be removed. This quickly cleans the dataset but may lead to loss of important information or bias if certain columns have many missing values.
2. **Impute data:** Replace null values with mean, median, or more complex estimations. While this retains important rows or columns, it introduces uncertainty as values are estimated.
3. **Mask the data:** Treat missing values as their own category, assuming they provide useful information. This method avoids losing rows or columns but adds uncertainty as all missing values are assumed to be alike.

Handling Outliers:



Outliers are distinct observations that deviate significantly from the rest of the data. While they can skew predictions, some outliers provide valuable insights. Here's how to address outliers:

1. **Identification:** Use plots like histograms, density plots, or box plots to visualize outliers. Additionally, consider standardized, deleted, or studentized residuals.

2. **Mathematical approach:** Calculate the interquartile range (IQR) using percentiles and define thresholds for outliers. Outliers fall outside the range defined by the IQR.
3. **Residuals:** Analyze residuals to detect outliers in the data.

Creating Plots:

- Use Seaborn's `displot` function for histograms and `boxplot` function for box plots.
- Pass data and appropriate arguments to generate the desired plot.

Mathematical Calculation for Outliers:

- Use numpy's `percentile` function to calculate percentiles.
- Define the IQR as the difference between the 75th and 25th percentiles.
- Determine thresholds for outliers using the IQR and percentiles.
- Identify outliers based on these thresholds.

Residuals and Approaches to Detect Outliers:

Residuals are the differences between the actual and predicted values from a model. They indicate model failure and can be leveraged to detect outliers. Here are some approaches:

1. **Standardized Residuals:** Divide the residual by the standard error to standardize it. This accounts for variations in outcome variable ranges.
2. **Deleted Residuals:** Remove the outlier observation from the dataset and observe the difference in model predictions compared to the original model.
3. **Studentized Residuals:** Similar to deleted residuals, but standardized according to the range of the model. It removes one observation at a time and evaluates the effect on the model.

Handling Outliers:

Once outliers are detected, several strategies can be employed:

1. **Remove Them:** Entirely eliminating outliers removes their effects but may lead to loss of important data.
2. **Replace with Different Value:** Assigning a different value to the outlier prevents its influence on the model but may still lose valuable information.
3. **Transform the Column:** Apply transformations such as log transformation to the column containing the outlier. This may mitigate its outlier status.
4. **Predict the Value:** Use similar observations or regression to predict what the outlier value would have been. However, this method may require significant effort and may still lose important information.
5. **Keep the Value:** Retain the outlier if using a model resistant to outliers.

1.6 Exploratory Data Analysis

EDA involves analyzing datasets to summarize their main characteristics, often using visual and statistical methods. It helps in understanding the dataset's structure, identifying patterns, and determining if further cleaning or additional data is needed.

Why is EDA Useful?

- EDA serves as an initial exploration of the dataset, akin to an introductory conversation with the data.

- It helps in identifying patterns and trends, which can be as important as the actual findings from modeling.

Summary Statistics in EDA:

- Average, median, minimum, maximum values.
- Correlations between different columns.
- Visualizations such as histograms, scatter plots, and box plots.

Tools for EDA:

- Pandas library for data wrangling and summary statistics.
- Matplotlib and Seaborn libraries for visualization.

Sampling in EDA:

- Random sampling from DataFrames can be useful for various reasons:
  - Reduce computation time for large datasets.
  - Train models on a subset of data and hold out another set for testing.
  - Ensure the sample is indicative of the proportions of different observations, especially for stratified sampling.

Code Example for Sampling:

```
# Assuming 'data' is a pandas DataFrame
sample = data.sample(n=5, replace=False)
print(sample.iloc[:, -3:])
```

- `sample`: Random sample of 5 rows from the DataFrame.
- `replace=False`: Ensures each row appears only once in the sample.
- `iloc[:, -3:]`: Selects the last three columns of the sample.
- Output displays the selected columns for the random sample.

Data Visualization Libraries:

1. **Matplotlib:**
   - Main library for creating plots and graphs in Python.
   - Offers flexibility and features for customization.
   - Foundation for visualization in Python.
2. **Pandas:**
   - Offers a convenient wrapper function around Matplotlib.
   - Allows for easier creation of plots directly from DataFrames.
   - Less flexible compared to Matplotlib but often sufficient for basic plotting needs.
3. **Seaborn:**
   - Built on top of Matplotlib.
   - Creates visually appealing plots with shorthand methods.
   - Offers statistical plot types such as linear model plots, pairwise correlation plots, etc.
   - Seaborn preferences are incorporated by Matplotlib upon import.

Creating Plots:

1. **Scatter Plot with Matplotlib:**
   - Use `plt.plot` for scatter plot.

    - Customize markers, colors, and labels.
    - Call `plt.legend` to add a legend.
2. **Multiple Layered Scatter Plots:**
    - Plot multiple scatter plots without reinitializing.
    - Customize colors and labels for each layer.
    - Add legend to differentiate between layers.
3. **Histogram with Matplotlib:**
    - Use `plt.hist` to plot histogram.
    - Specify the number of bins for the distribution.
4. **Customizing Plots with Object-Oriented Syntax:**
    - Use `plt.subplot` to create separate figures and axes.
    - Customize ticks, labels, title, etc., using `ax.set`.
5. **Pandas Syntax for Plotting:**
    - Group data using `groupby` and compute statistics.
    - Call `.plot` directly on the DataFrame.
    - Customize plot parameters such as colors, labels, and figure size.
6. **Pair Plot and Hexbin Plot with Seaborn:**
    - Use `sns.pairplot` for scatter plot matrix.
    - Customize plot appearance and add hue for categorical differentiation.
    - Use `sns.jointplot` for hexbin plot, which shows the density of data points.
7. **Facet Grid with Seaborn:**
    - Use `sns.FacetGrid` for plotting across different categories.
    - Customize row or column variables to split the data.
    - Use `.map` to apply different plots to each subset.

1.7 Feature Engineering and Variable Transformation

Feature Engineering:

- Process of creating new features or modifying existing ones to improve model performance.
- Involves extracting meaningful information from raw data.

Variable Transformation:

- Modifying variables to meet assumptions of machine learning algorithms.
- Examples include log transformations to address outliers and ensure normal distribution.

Feature Encoding:

- Categorical variables need to be converted into numerical format for modeling.
- Techniques like one-hot encoding and label encoding are commonly used.

Feature Scaling:

- Ensures consistency in scale across different features.
- Common scaling techniques include Min-Max scaling and Standardization.

Linear Regression Assumptions:

- Linear regression assumes a linear relationship between predictor variables and the target variable.

- Example: $y_\beta(x) = \beta_0 + \beta_1 x_1 + \beta_2 x_2$
- Parameters $\beta = (\beta_0, \beta_1, \beta_2)$ are learned by the model.

Example:

- Consider predicting box office returns with features like cast (x_1) and marketing budget (x_2)
- Parameters β_1, β_2 determine the impact of each feature on revenue prediction.

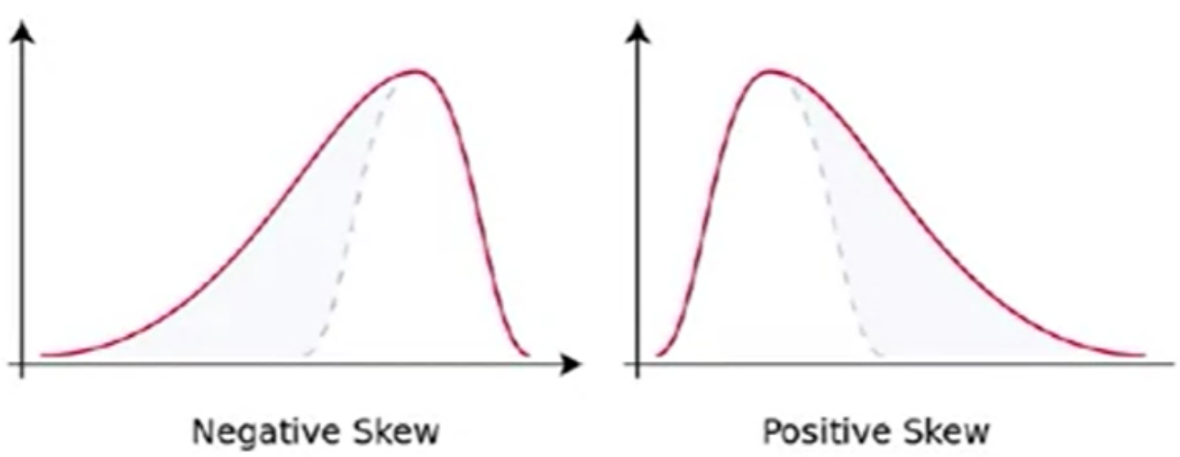
Transformations for Linear Relationships:

- Transforming variables (x_1, x_2) to ensure a linear relationship with the target variable.
- Even after transformation, the relationship remains linear: $y = \beta_0 + \beta_1 f(x_1) + \beta_2 g(x_2)$

1.8 Data Transformation and Feature Engineering

Normality Assumption in Linear Regression:

- Linear regression assumes that residuals from the model are normally distributed.
- However, raw data may often be positively or negatively skewed, which violates this assumption.



Data Transformation:

- Data transformations help algorithms find better solutions by ensuring residuals are normally distributed.
- Transformation techniques include logarithmic transformations and Box-Cox transformations.

Logarithmic Transformation:

- Logarithmic transformations are applied to positively skewed data.
- They help in achieving a more normal distribution of data.
- $log(x)$ is used to transform positively skewed data to a more normal distribution.

Linear Regression with Transformed Features:

- Transformed features maintain a linear relationship with the target variable.
- Although features are transformed, the underlying model remains linear.

Example:

- In cases where a linear relationship may not exist due to diminishing returns, logarithmic transformations help.
- For instance, larger budgets may not have a linear relationship with box office returns, but a logarithmic relationship may exist.

Polynomial Features:

- Polynomial features help capture higher-order relationships in data.
- They add flexibility to the model by incorporating features like x^2, x^3, etc.
- Despite including polynomial features, the model remains linear.

Example:

- Adding polynomial features like x^2 to the budget feature captures curvature in the relationship.
- Higher-order polynomial features capture multiple inflection points, indicating complex relationships.

Implementation with scikit-learn:

- `PolynomialFeatures` from `sklearn.preprocessing` is used to create polynomial transformations.
- It creates transformer objects that can fit to and transform data, allowing the generation of polynomial features.

Variable Selection:

- Variable selection involves choosing the features to include in the model.
- Features often need to be transformed before being included in models.

Variable Transformation:

- Transformation methods include logarithmic and polynomial transformations.
- These transformations help meet assumptions of machine learning algorithms.

Encoding:

- Encoding involves converting non-numeric features (categorical or ordinal) into numeric features.

Types of Encoding:

1. **Binary Encoding:**
   - Suitable for variables with two possible values (e.g., male/female, true/false).
   - Converts variables to 0 or 1.
2. **One-Hot Encoding:**
   - Suitable for variables with multiple values.
   - Creates binary variables for each category.
   - Each category becomes a column with values true (1) or false (0).
3. **Ordinal Encoding:**
   - Converts ordered categories into numerical values.
   - Assigns integers based on the order of categories (e.g., low, medium, high -> 1, 2, 3).
   - Assumes a linear relationship between categories, which may not always be appropriate.

Considerations:

- Ordinal encoding assigns a numerical order to categories, which may imply a linear relationship.
- One-hot encoding preserves categorical information but increases dimensionality.
- The choice of encoding method depends on the nature of the data and the requirements of the model.

Implementation:

- Encoding methods are applied based on the type of feature and the desired representation in the model.
- One-hot encoding and ordinal encoding are common techniques used in practice.

1.9 Feature Scaling

Definition:

- Feature scaling involves adjusting the scale of variables to allow for comparison of variables with different scales.

Importance:

- In real-world data, continuous features often have different scales.
- Scaling helps algorithms interpret features correctly and improves model performance.

Example:

- Consider two features: product price (0-10) and number of stores selling the product (10,000-50,000).
- Bringing these features to similar scales enables fair comparison.

Issues with Different Scales:

- Using algorithms like K Nearest Neighbors (KNN) can be affected by scale discrepancies.
- Extreme differences in scale can lead to incorrect grouping or classification.
- For example, age measured in seconds versus number of surgeries.

Scaling Methods:

1. **Standard Scaling (Z-score normalization):**
    - Subtract the mean and divide by the standard deviation.
    - Measures spread in standardized units.
    - Affected by outliers but less so than Min-Max scaling.
2. **Min-Max Scaling:**
    - Rescales values to fit within a specified range (e.g., 0 to 1).
    - Subtracts the minimum value and divides by the range (max - min).
    - Sensitive to outliers and can compress data if outliers are present.
3. **Robust Scaling:**
    - Similar to Min-Max scaling but focuses on the interquartile range.
    - Robust to outliers and avoids skewing caused by extreme values.
    - Does not ensure values remain between 0 and 1.

Considerations:

- The choice of scaling method depends on the nature of the data and the algorithm being used.
- Standard scaling is robust but sensitive to outliers.
- Min-Max scaling is simple but can be influenced by outliers.
- Robust scaling is a good compromise for handling outliers without compromising the scale range.

Conclusion:

- Feature scaling is essential for ensuring fair comparison of variables with different scales.
- Different scaling methods offer trade-offs between simplicity, robustness, and sensitivity to outliers.
- The choice of scaling method should be based on the specific requirements of the model and the characteristics of the data.

Variable Transformations:

1. **Continuous Numerical Values:**
   - Use transformations like Standard Scaling, Min-Max Scaling, and Robust Scaling.
   - Implementations available in `sklearn.preprocessing`: `StandardScaler`, `MinMaxScaler`, and `RobustScaler`.
2. **Nominal or Categorical Data (Unordered):**
   - Use binary encoding (for binary features) or one-hot encoding (for multiple categories).
   - Functions available in `sklearn.preprocessing`: `LabelEncoder`, `LabelBinarizer`, and `OneHotEncoder`.
   - `get_dummies` from Pandas can also perform one-hot encoding.
3. **Ordinal Data (Ordered Categorical):**
   - Encode ordinal variables with numerical values representing the order.
   - Use `DictVectorizer` or `OrdinalEncoder` from `sklearn.preprocessing`.

1.10 Estimation and Inferences

Customer Churn Data Analysis:

- **Dataset:** Telco customer churn dataset from IBM Cognos Analytics.
- **Variables:** Includes account type, customer characteristics, revenue per customer, customer satisfaction score, estimated lifetime value, churn status, and churn type.
- **DataFrame:** The data for the phone subscription is contained in a Pandas DataFrame assigned to the variable `df_phone`.

Exploratory Data Analysis (EDA):

1. **Bar Plot:** Shows churn likelihood based on payment types. Customers using credit cards are less likely to churn compared to those using bank withdrawal payments or mailed checks.
2. **Bar Plot (Months vs. Churn):** Utilizes `pd.cut` to categorize customer tenure into five equal-length bins. Shows that customers with shorter tenure are more likely to churn.
3. **Pair Plot:** Displays pairwise relationships between variables such as customer tenure, gigabytes used per month, total revenue, customer lifetime value, and churn status. Split by churn value (1 for churned, 0 for not churned).
4. **Hexbin Plot:** Joint plot showing tenure in months versus monthly charge. Helps visualize the relationship between customer tenure and monthly charge, with hexbins indicating

density of data points.

Parametric vs. Non-parametric Statistics:

- **Statistical Inference:** Involves understanding the underlying data generating process and modeling the distribution of data.
- **Parametric Model:** Constrained to a finite number of parameters and relies on strict assumptions about the distribution of the data.
- **Non-parametric Model:** Inference does not depend on specific distributional assumptions; it's distribution-free and relies more on the available data.
- **Example of Non-parametric Inference:** Creating a histogram to visualize the data distribution without assuming a specific distribution.
- **Parametric Model Example:** Linear regression, which assumes a linear relationship between variables and a normally distributed error term.
- **Parameter Estimation in Parametric Models:** Often done through Maximum Likelihood estimation, where parameters are chosen to maximize the likelihood function given the sample data.
- **Common Statistical Distributions:**
    1. **Uniform Distribution:** Equal probability for all values within a range.
    2. **Normal (Gaussian) Distribution:** Bell-shaped curve with most values around the mean and equal likelihood of values symmetrically distributed around the mean.
    3. **Log-normal Distribution:** Result of taking the logarithm of a skewed variable to achieve a more normal distribution.
    4. **Exponential Distribution:** Describes the time between events occurring continuously and has most values near zero.
    5. **Poisson Distribution:** Models the number of events occurring in a fixed interval of time or space.

Central Limit Theorem:

- States that the distribution of sample means from a large number of samples will approximate a normal distribution, regardless of the original distribution of the population.
- Commonly observed in real-world phenomena such as human height, where most individuals cluster around the average height.

Business Example - Customer Lifetime Value:

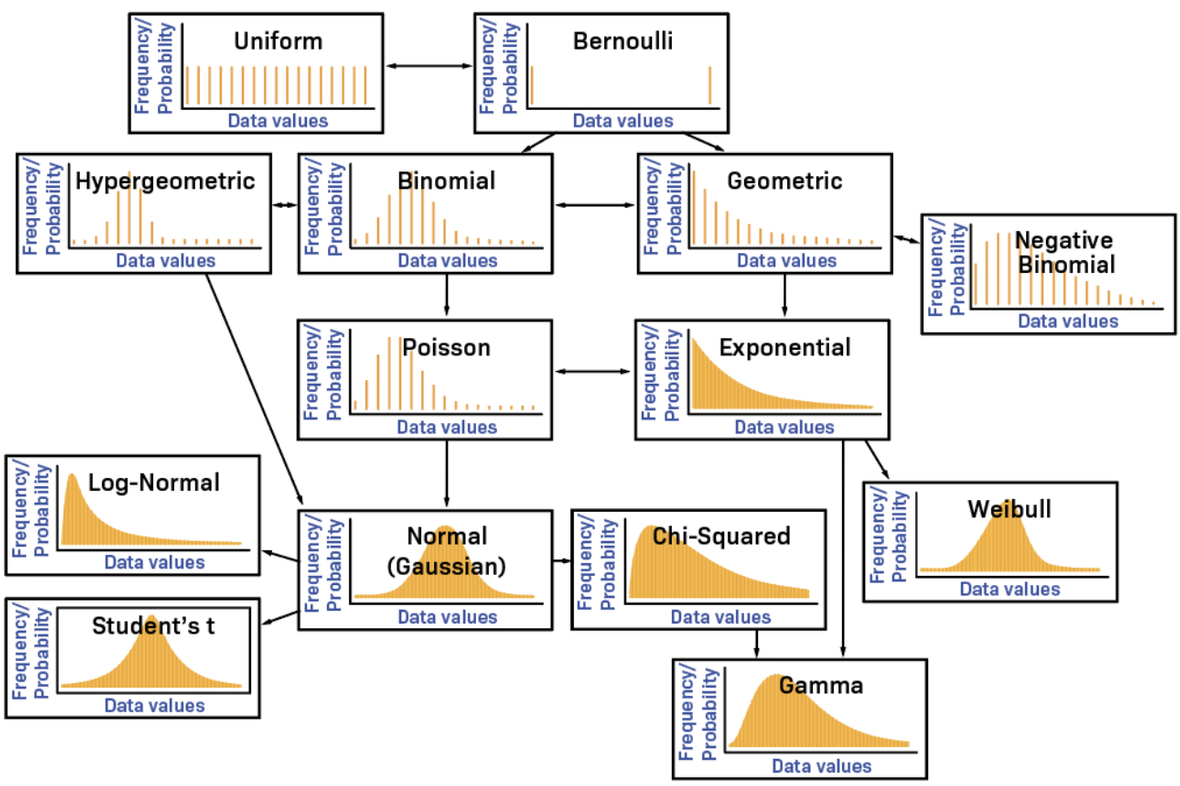
- Estimating customer lifetime value involves assumptions about the expected duration of customer tenure and their spending over time.
- Assumptions can be parametric (based on specific distributions) or non-parametric (relying more on observed data).
- Parametric models often use Maximum Likelihood estimation to determine the parameters that maximize the likelihood function given the data.

Understanding parametric and non-parametric statistics is essential for choosing appropriate modeling techniques and making accurate inferences from data.

Common Statistical Distributions:

1. **Uniform Distribution:**
    - Equally likely chance for all values within a range.
    - Example: Rolling a fair six-sided die, where each number has an equal probability of occurring.

2. **Normal (Gaussian) Distribution:**
    - Bell-shaped curve where most values cluster around the mean.
    - Equal likelihood for values symmetrically distributed around the mean.
    - Parameters: Mean defines the center, standard deviation defines the spread.
    - Central Limit Theorem: Distribution of sample means approximates a normal curve with enough samples.
    - Real-world example: Human height distribution, where most individuals are close to the average height.
3. **Log-normal Distribution:**
    - Result of taking the logarithm of a skewed variable to achieve a more normal distribution.
    - Tighter spread around the mean results in a distribution closer to normal.
    - Common in real-world scenarios involving skewed data, such as income distribution.
4. **Exponential Distribution:**
    - Describes the time between events occurring continuously.
    - Most values cluster near zero, representing shorter time intervals.
    - Used to model scenarios like the time between occurrences of events.
5. **Poisson Distribution:**
    - Models the number of events occurring in a fixed interval of time or space.
    - Parameter Lambda (λ) represents both the average and variance of the distribution.
    - Example: Predicting the number of customers visiting a website in a given time interval.



Understanding these common distributions is crucial for analyzing and modeling real-world data across various domains. Each distribution has its unique characteristics and applications, allowing statisticians and data scientists to make informed decisions and predictions based on data observations.

Difference Between Frequentist and Bayesian Statistics:

1. **Frequentist Statistics:**

    - Concerned with repeated observations approaching infinity.
    - Starts without prior knowledge of probabilities.
    - Estimates probabilities based on many repeats of an experiment.
    - Derives estimates directly from data with no external influence.
    - Confidence in estimates based on the size of the sample.
    - Fixed value for probability in the population.
    - Interpretation focuses on estimating population parameters based on data observations.
2. **Bayesian Statistics:**
    - Parameters themselves can have a probability distribution.
    - Incorporates prior beliefs or knowledge about the parameters.
    - Allows experimenters to start with educated guesses and update beliefs based on data.
    - Uses probability distributions for parameters.
    - Updates prior distribution after observing data to obtain posterior distribution.
    - Tighter posterior distribution with more data.
    - Interpretation focuses on updating beliefs about parameters based on observed data.

Both frequentist and Bayesian approaches use similar mathematical techniques but differ in their interpretation and handling of uncertainty. Frequentist statistics relies solely on data observations to estimate population parameters, while Bayesian statistics incorporates prior beliefs and updates them based on observed data. The choice between the two approaches depends on the nature of the problem and the availability of prior information.

1.11 Hypothesis Testing

Hypothesis testing involves making statements about population parameters.
Example: Estimating the mean of a Poisson distribution or the number of people entering a grocery store line per hour.

Types of Hypotheses:

- Null Hypothesis (H_0): Represents a specific value or condition.
- Alternative Hypothesis (H_a or H_1): Represents a different value or condition from the null.

Setting Up Hypotheses:

- Null hypothesis is typically the specific value or condition being tested.
- Alternative hypothesis is generally less specific, representing values different from the null.

Hypothesis Testing Procedure:

- Utilizes data from a sample to make a decision regarding the hypotheses.
- Decide whether to accept the null hypothesis or reject it in favor of the alternative.

Acceptance and Rejection:

- Commonly stated as "reject the null hypothesis but never accept the alternative."
- In practical terms, accepting the alternative hypothesis based on the test statistic.

Bayesian Interpretation:

- Provides posterior probabilities for both null and alternative hypotheses.
- Determines which hypothesis is more likely based on the posterior probabilities.

Hypothesis testing is a fundamental concept in statistics used to make decisions about population parameters based on sample data, with the aim of determining whether evidence supports or contradicts a specific hypothesis.

Coin Tossing Example:

1. **Scenario Setup:**
   - Two coins: Coin 1 (70% chance of heads) and Coin 2 (50/50 chance of heads).
   - Objective: Determine which coin is more likely to be picked based on the outcome of 10 tosses.
2. **Experimental Procedure:**
   - Select one coin randomly without inspecting.
   - Toss the chosen coin 10 times and count the number of heads obtained.
3. **Probability Calculation:**
   - Calculate the probability of obtaining a specific number of heads for both coins.
   - Create a table displaying the probabilities of heads for each coin based on the number of tosses.
4. **Likelihood Ratio:**
   - Calculate the likelihood ratio based on the observed number of heads.
   - Example: If three heads are observed, calculate the probability of obtaining three heads for both coins.
   - Coin 1: Probability = 0.117
   - Coin 2: Probability = 0.009
   - Likelihood ratio: Coin 1 was 13 times more likely to produce three heads than Coin 2.
5. **Interpretation:**
   - Likelihood ratio represents the ratio of how likely one hypothesis (e.g., Coin 1) is compared to another (e.g., Coin 2) based on the observed data.

By comparing the likelihood ratios, we can determine which hypothesis (coin) is more likely to have produced the observed outcome, thus making an informed decision about which coin is more probable to have been selected initially.

Bayesian Interpretation of Hypothesis Testing:

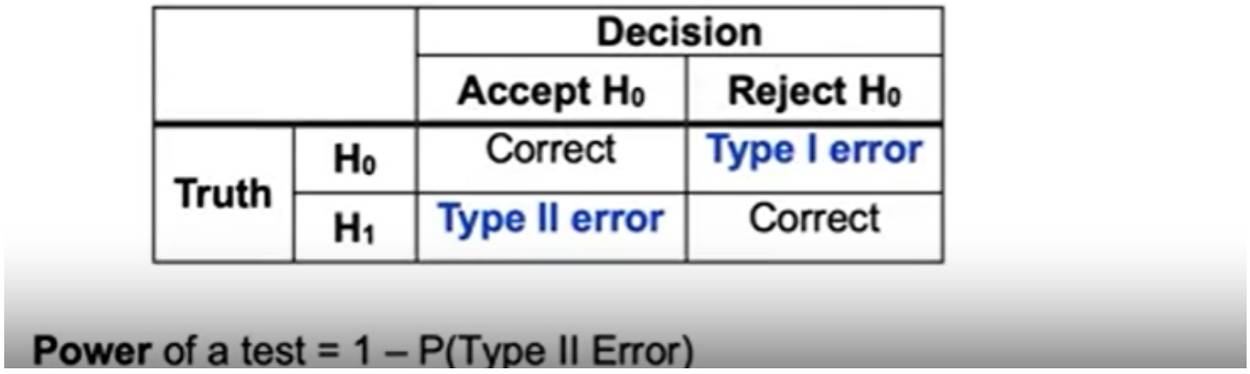
1. **Priors for Hypotheses:**
   - We assign prior probabilities to each hypothesis
   - In the coin toss example, both hypotheses (0.5 and 0.7) are given equal prior probabilities (0.5 each).
2. **Significance of Priors:**
   - Priors represent our initial beliefs about the likelihood of each hypothesis before observing data.
   - In the absence of prior knowledge, equal priors are often assumed.
   - Priors influence the posterior distribution, which is our updated belief after observing data.
3. **Bayes' Rule Application:**
   - Bayes' rule is used to update our prior beliefs based on observed data.
   - The posterior distribution represents the updated probability of each hypothesis after considering the data.
4. **Calculation of Posterior Distribution:**
   - The posterior distribution is calculated by multiplying the prior distribution with the likelihood ratio.

- Likelihood ratio represents the relative likelihood of the observed data under each hypothesis.

5. **Role of Likelihood Ratio:**
   - Likelihood ratio quantifies the evidence provided by the data in favor of one hypothesis over another.
   - It does not depend on the prior probabilities.
6. **Updating Priors:**
   - Priors are updated based on the likelihood ratio, resulting in the posterior distribution.
   - Posterior distribution reflects our updated beliefs about the hypotheses after observing the data.

In summary, the Bayesian approach to hypothesis testing incorporates prior beliefs about hypotheses and updates them based on observed data, resulting in the posterior distribution. This allows for a probabilistic interpretation of hypotheses and accounts for uncertainty in decision-making.

Type I and Type II Errors:



1. **Type I Error:**
   - Definition: Incorrectly rejecting the null hypothesis.
   - Example: In the coin toss scenario, concluding that the coin is biased when it is actually fair.
   - Interpretation: False positive; detecting an effect or relationship that does not exist.
2. **Type II Error:**
   - Definition: Incorrectly accepting the null hypothesis.
   - Example: In the customer churn prediction, failing to identify that customers with longer tenure are less likely to churn.
   - Interpretation: False negative; failing to detect an effect or relationship that does exist.
3. **Power of a Test:**
   - Definition: The probability of correctly rejecting the null hypothesis when it is false.
   - Interpretation: A high-power test minimizes the likelihood of a Type II error by effectively detecting true effects.

Examples of Hypothesis Testing in the Workplace:

1. **Customer Churn Prediction:**
   - Null Hypothesis: Customer churn is due to chance.
   - Alternative Hypothesis: Customers with longer tenure are less likely to churn.
   - Type I Error: Incorrectly concluding that tenure does affect churn when it does not.
   - Type II Error: Failing to identify the relationship between tenure and churn when it exists.

2. **Product Testing:**
    - Null Hypothesis: There is no difference in product performance between two versions.
    - Alternative Hypothesis: One version performs better than the other.
    - Type I Error: Incorrectly concluding that one version outperforms the other when it does not.
    - Type II Error: Failing to detect a performance difference between versions when one exists.
3. **Market Research:**
    - Null Hypothesis: There is no correlation between advertising expenditure and sales.
    - Alternative Hypothesis: Increased advertising expenditure leads to higher sales.
    - Type I Error: Incorrectly concluding that advertising expenditure influences sales when it does not.
    - Type II Error: Failing to identify the impact of advertising expenditure on sales when it exists.

In summary, understanding and minimizing the risks of Type I and Type II errors are crucial in hypothesis testing, especially in practical scenarios where decisions are made based on statistical analyses.

Important Terminology in Hypothesis Testing:

1. **Test Statistic:**
    - Definition: A value calculated from sample data that is used to determine whether to accept or reject the null hypothesis.
    - Example: Likelihood ratio in coin flipping, difference in means between two groups in t-tests.
2. **Rejection Region:**
    - Definition: The set of values for the test statistic for which the null hypothesis is rejected.
    - Interpretation: Values falling within the rejection region indicate evidence against the null hypothesis.
3. **Acceptance Region:**
    - Definition: The set of values for the test statistic for which the null hypothesis is accepted.
    - Interpretation: Values falling within the acceptance region suggest insufficient evidence to reject the null hypothesis.
4. **Null Distribution:**
    - Definition: The distribution of the test statistic assuming the null hypothesis is true.
    - Interpretation: Helps determine the likelihood of observing certain values of the test statistic under the null hypothesis.

Business Examples of Hypothesis Testing:

1. **Marketing Intervention Impact:**
    - Null Hypothesis: The marketing campaign has no impact on purchasing behavior.
    - Alternative Hypothesis: The marketing campaign has a significant impact on purchasing behavior.
2. **Website Layout Change Impact:**
    - Null Hypothesis: The change in website layout does not affect website traffic.
    - Alternative Hypothesis: The change in website layout significantly affects website traffic.

3. **Product Quality Assurance:**
    - Null Hypothesis: Product size is not significantly different from the expected size.
    - Alternative Hypothesis: Product size deviates significantly from the expected size.

Significance Levels and P-values:

Significance Level (α):

- Definition: The predetermined threshold for rejecting the null hypothesis.
- Interpretation: Lower α values indicate a higher threshold for accepting evidence against the null hypothesis, reducing the risk of Type I errors.

P-values:

- Definition: The probability, under the null hypothesis, of obtaining a result as extreme as or more extreme than the observed data.
- Interpretation: Lower P-values suggest stronger evidence against the null hypothesis.

Setting α:

- Importance: α should be chosen before conducting the hypothesis test to avoid P-hacking and ensure the reliability of results.
- Example: Lower α values (e.g., 0.01 or 0.05) are suitable for critical decisions (e.g., medication effectiveness), while higher values may suffice for less critical decisions (e.g., font size changes in advertisements).

Interpretation of P-values:

- Definition: P-value represents the probability of observing the data under the assumption that the null hypothesis is true.
- Significance: A small P-value indicates that the observed data is unlikely to occur if the null hypothesis were true, leading to rejection of the null hypothesis.

Confidence Intervals:

- Definition: Range of values within which the null hypothesis is accepted.
- Interpretation: Complementary to P-values, confidence intervals provide an interval estimate for the null hypothesis.

Example:

- Coin Tossing Scenario:
    - Null Hypothesis: The coin is fair (P(heads) = 0.5).
    - Alternative Hypothesis: The coin is unfair (P(heads) ≠ 0.5).
    - Test: If observing three heads in ten flips.
    - Calculation: Using the binomial distribution, calculate the cumulative probability of observing three heads or fewer.
    - Decision: If the calculated P-value exceeds the significance level (e.g., 0.05), fail to reject the null hypothesis.

F-Statistic:

- **Null Hypothesis for the F-Statistic:** The null hypothesis states that the data can be modeled by setting all regression coefficients (betas) to zero. In the context of linear

regression, this implies that none of the independent variables have an effect on the dependent variable.

- **Interpretation of the F-Statistic:** The F-statistic is a test statistic used in regression analysis to assess the overall significance of the regression model. It evaluates whether the addition of independent variables improves the model's predictive power compared to using just the mean of the dependent variable.
- **Rejection of the Null Hypothesis:** A small p-value associated with the F-statistic indicates that the observed data is unlikely to occur under the assumption of no effects (all betas equal to zero). Therefore, the null hypothesis is rejected in favor of the alternative hypothesis, suggesting that at least one independent variable has a significant effect on the dependent variable.

Power and Sample Size:

- **Type I Error and Multiple Tests:** Conducting multiple hypothesis tests increases the probability of committing at least one Type I error (incorrectly rejecting the null hypothesis when it is true). The likelihood of Type I errors accumulates as the number of tests increases.
- **Bonferroni Correction:** To control the overall Type I error rate when conducting multiple tests, the Bonferroni Correction adjusts the significance level (alpha) for each individual test. The corrected significance level is determined by dividing the desired overall Type I error rate by the number of tests. This correction helps maintain the overall error rate at the desired level but may reduce the power of individual tests.
- **Trade-off Between Type I Error and Power:** Implementing the Bonferroni Correction reduces the likelihood of Type I errors but may decrease the power of the tests. Larger effects or larger sample sizes may be required to detect significant results after applying the correction.

Best Practices:

- **Limiting Comparisons:** It is advisable to limit the number of comparisons to a few well-motivated cases to avoid excessive testing and potential inflation of Type I error rates.

Correlation vs. Causation:

- Correlation: Describes the statistical relationship between two variables, indicating how they tend to change together.
- Causation: Implies a cause-and-effect relationship, where changes in one variable directly cause changes in another.

Understanding Correlation:

- Correlated variables may have predictive power but do not necessarily imply causation.
- Examples:
  - Rainfall may be correlated with cooler temperatures, but the causal mechanism varies based on geographical factors.
  - Correlation does not imply directionality or causality.

Confounding Variables:

- Confounding variables are external factors that influence both the independent and dependent variables, leading to a correlation between them.
- Examples:

    - Population size may confound the correlation between car accidents and the number of people named John.
    - Temperature may confound the correlation between ice-cream sales and drownings.

Spurious Correlations:

- Spurious correlations are coincidental relationships between variables that do not have a causal connection.
- Examples:
    - High correlation between the age of Miss America winners and deaths caused by steam, hot vapors, and hot objects.
    - Strong correlation between worldwide non-commercial space launches and sociology doctorate awards.
- Spurious correlations may arise due to random chance or shared underlying trends over time.

Implications:

- Understanding the difference between correlation and causation is crucial for making informed decisions and avoiding misinterpretation of data.
- Consideration of confounding variables helps in identifying the true causal relationships between variables.
- Awareness of spurious correlations helps in critically evaluating statistical findings and avoiding drawing erroneous conclusions.