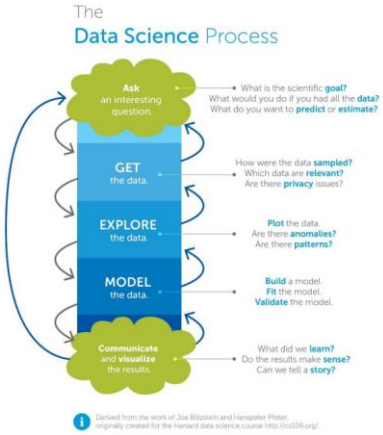
Contents

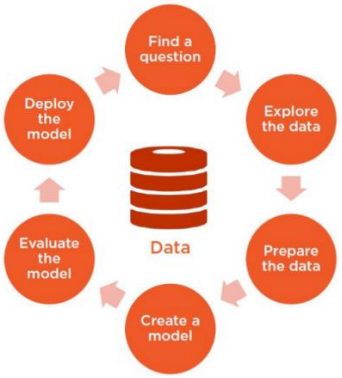
1. Introduction to Data Science

Data Science Definitions

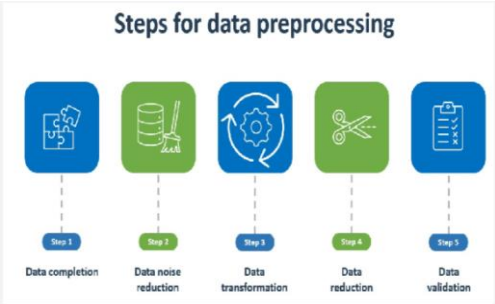
- Set of fundamental principles that guide the extraction of knowledge from data.
- Field of study that combines domain expertise, programming skills, and knowledge of maths and statistics to extract meaningful insights from data.

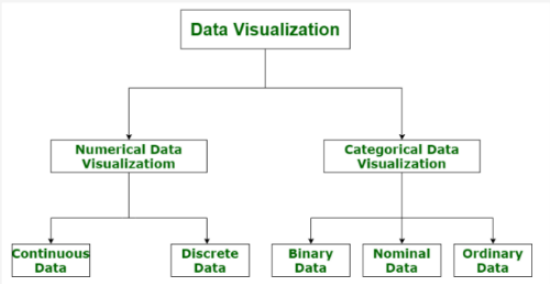
The Data Science Process





Data Cleaning Techniques and EDA





1. Numerical Data Visualization

Numerical data refers to data that is quantifiable and can be measured. It can be further divided into:

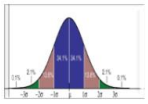
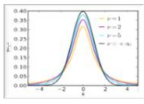
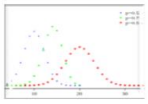
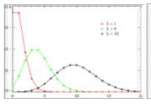
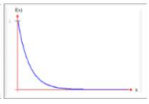
- Continuous Data: Data that can take any value within a range. For example, height, weight, temperature, or time. Continuous data is often visualized using line charts, histograms, or scatter plots.
- Discrete Data**:** Data that can only take specific, distinct values, often counted in whole numbers. Examples include the number of people, cars, or objects. Discrete data is commonly visualized with bar charts or dot plots.

2. Categorical Data Visualization

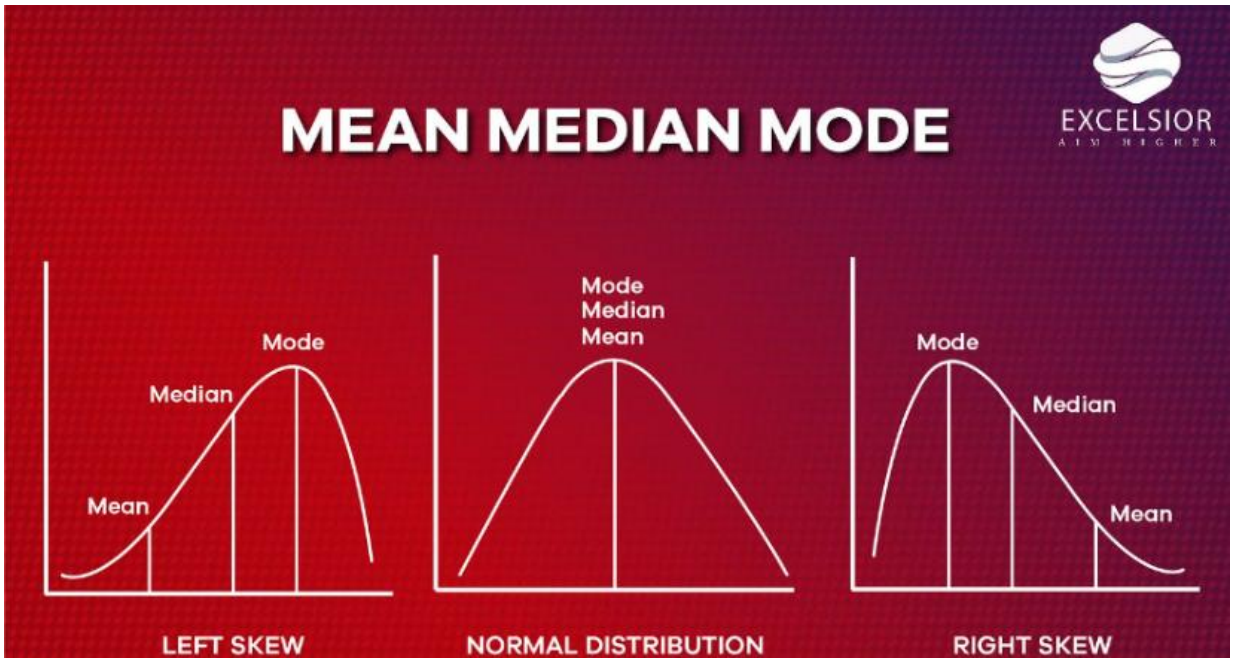
Categorical data is used to represent characteristics or attributes that can be grouped into categories. It is further divided into:

- Binary Data**:** Data with only two possible values, often represented as 0 and 1 or yes and no. Binary data can be visualized using bar charts or pie charts.
- Nominal Data: Data that represents categories without a specific order. Examples include types of fruit, colors, or countries. Nominal data is often visualized with pie charts or bar charts.
- Ordinal Data: Data with categories that have a meaningful order or ranking, such as levels of satisfaction (e.g., poor, average, good, excellent) or educational qualifications. Ordinal data can be visualized with bar charts or ordered dot plots to reflect the ranking.

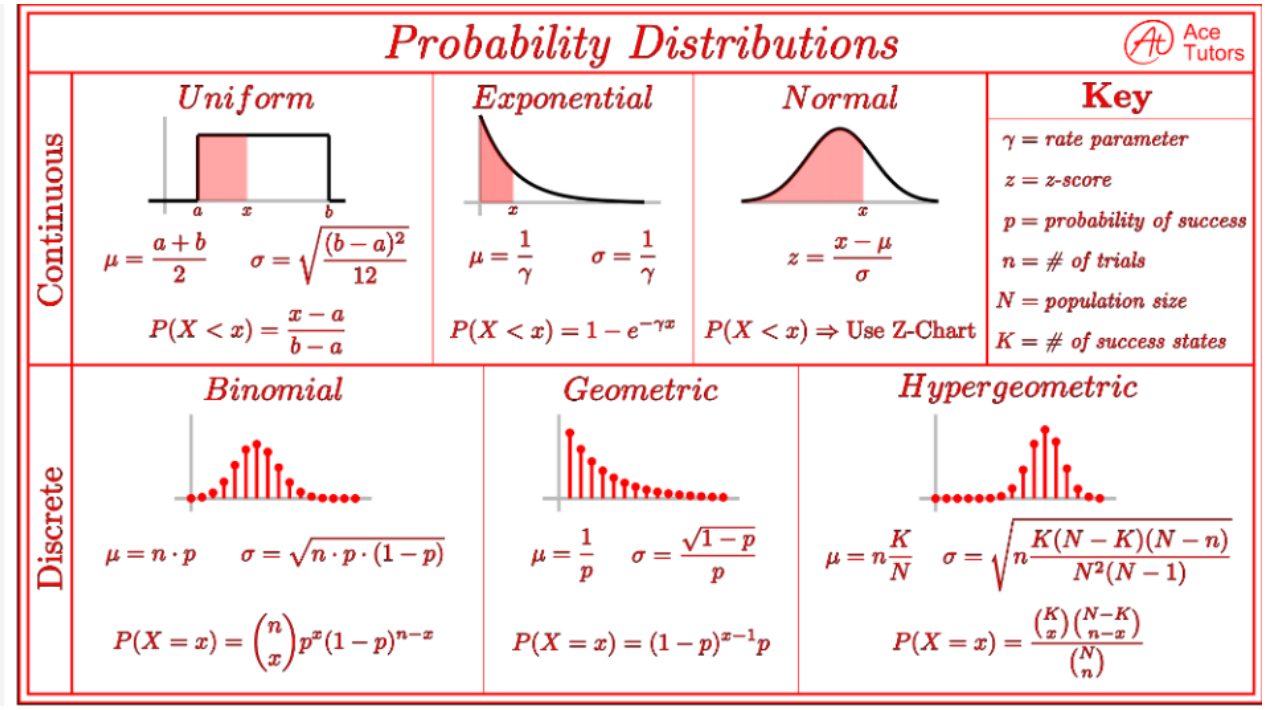
Data Distributions

	Normal Distribution	Student's T-Distribution	Binomial Distribution	Poisson Distribution	Exponential Distribution
What does it look like?					
Defining Characteristics	Distinctive Bell Shape	Shorter, fatter than the normal distribution.	Two outcomes: Success/Failure	Various shapes, but valid only for integers on the x-axis.	Models Time Between Events
Example of When to Use It	Modeling natural phenomena (height, weight, IQ, test scores etc.)	When you have small samples or don't know the population variance (σ^2).	Coin Toss Probability (Heads, Tails)	Gives probability of number of events in a fixed interval.	"How much time will go by before a major hurricane hits the Atlantic Seaboard?"
Example of DS Application	Least squares fitting or propagation of uncertainty.	Unknown σ^2 is common in real life data, you you'll have to use the T instead of the normal in that case.	Anywhere where binary (yes/no, black/white, vote/don't vote) data is used.	Anywhere there is a waiting time between events.	Building continuous-time Markov chains.

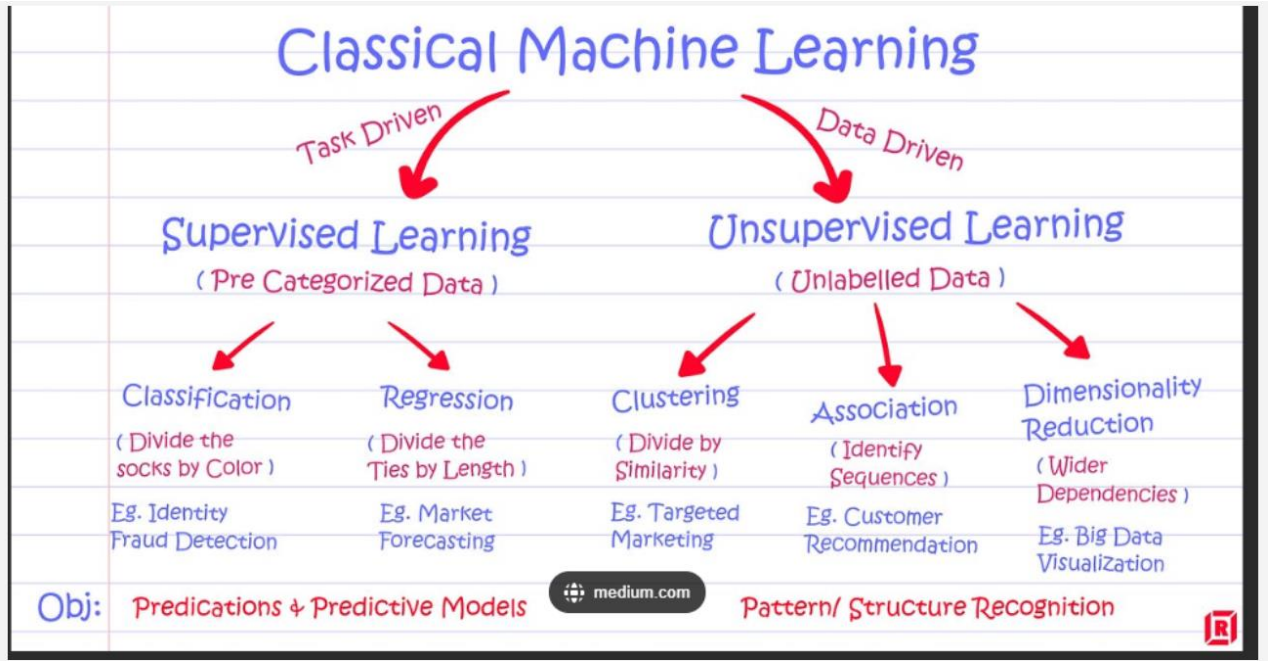
Statistical Distribution



Probability Distribution



Introduction to Machine Learning (Regression, Classification, Clustering and Decision Tree)



Model Evaluation Techniques

	Actual Positive	Actual Negative
Predicted Positive	True Positive(TP)	False Positive(FP) (Type 1 Error)
Predicted Negative	False Negative(FN) (Type 2 Error)	True Negative(TN)

$$\text{Accuracy} = \frac{\text{True Positive + True Negative}}{\text{Total Population}}$$

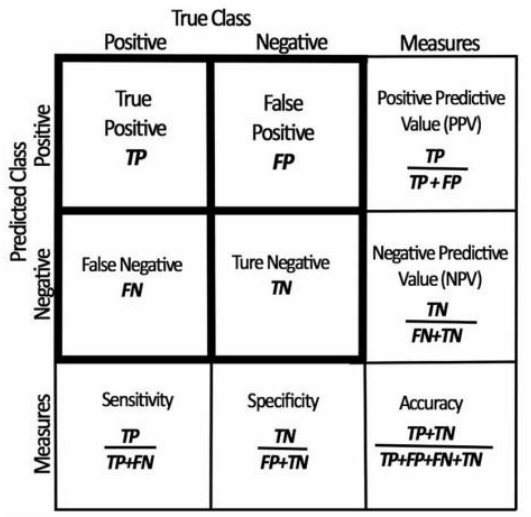
$$\text{Error Rate/Misclassification rate} = \frac{\text{False Positive + False Negative}}{\text{Total Population}}$$

$$\text{Precision} = \frac{\text{True Positive}}{\text{Predicted Positive(TP+FP)}}$$

$$\text{Sensitivity/Recall} = \frac{\text{True Positive}}{\text{Actual Positive(TP+FN)}}$$

$$\text{Specificity} = \frac{\text{True Negative}}{\text{Actual Negative(FP+TN)}}$$

$$\text{F1 Score} = \frac{2 * (\text{Recall} * \text{Precision})}{\text{Recall + Precision}}$$



Confusion Matrix Components

- True Positive (TP): The model correctly predicts a positive outcome. For example, it correctly predicts that a person has a disease.
- True Negative (TN): The model correctly predicts a negative outcome. For example, it correctly predicts that a person does not have a disease.
- False Positive (FP): The model incorrectly predicts a positive outcome. This is also known as a Type 1 Error. For example, it predicts that a person has a disease when they do not.
- False Negative (FN): The model incorrectly predicts a negative outcome. This is also known as a Type 2 Error. For example, it predicts that a person does not have a disease when they do.

Model Performance Matrices

1. Accuracy is the ratio of correct predictions (TP + TN) to the total number of predictions. It shows how often the model is correct overall.
2. The error rate is the proportion of incorrect predictions (FP + FN) to the total population. It indicates how often the model makes mistakes.
3. Precision measures the accuracy of positive predictions. It is the ratio of true positives to all predicted positives (TP + FP). Precision is important when the cost of false positives is high.
4. Recall, or sensitivity, measures the model's ability to identify all actual positives. It is the ratio of true positives to all actual positives (TP + FN). Recall is critical when the cost of false negatives is high.

5. Specificity measures the model's ability to correctly identify negatives. It is the ratio of true negatives to all actual negatives (FP + TN). This metric is important when it's crucial to minimize false positives.
6. F1 score is the harmonic mean of precision and recall. It balances the two metrics, especially useful when there is an uneven class distribution or when both false positives and false negatives are costly.

2. Introduction to Data Mining and Big Data

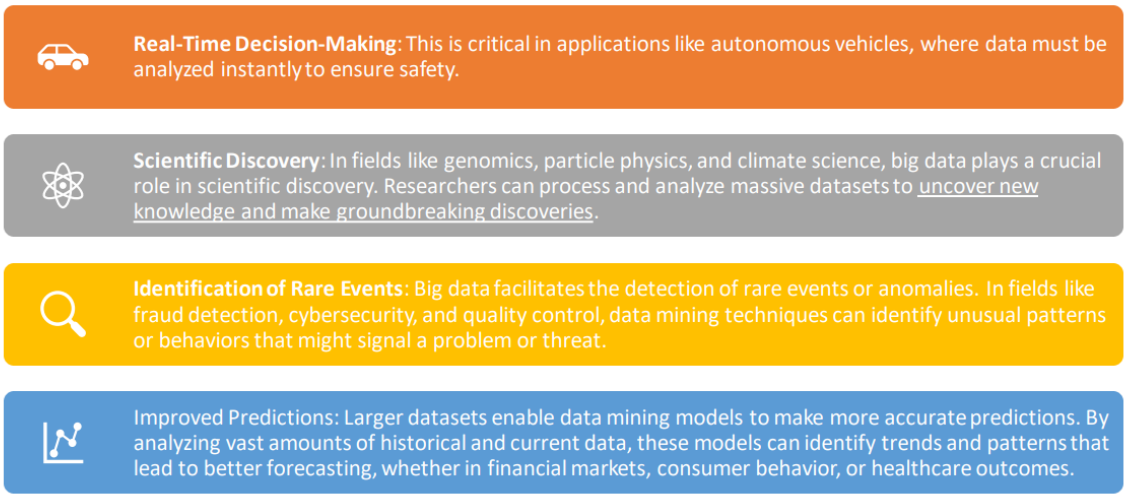
What is Big Data: Big Data refers to extremely large and complex datasets that are beyond the capability of traditional data-processing software to handle efficiently. Big Data encompasses not only the volume of data but also the speed at which it is generated and the variety of forms it takes. These massive datasets require advanced tools, technologies, and methods to store, process, and analyze to extract valuable insights. Organizations use Big Data to gain insights into patterns, trends, and associations that would be difficult to detect otherwise.

"4 V's"

1. **Volume**: Refers to the vast amount of data generated every second. Examples include social media posts, online transactions, sensor data from IoT devices, and satellite imagery. The volume of data can be in terabytes, petabytes, or even zettabytes.
2. **Velocity**: Refers to the speed at which data is generated, collected, and processed. For example, millions of messages, likes, and transactions are created every second on social media platforms. Big Data systems are often required to analyze data in real-time or near real-time, especially for applications like fraud detection, personalized advertising, and predictive maintenance.
3. **Variety**: Big Data includes a wide range of data types, such as structured (e.g., databases), semi-structured (e.g., JSON files, XML), and unstructured data (e.g., images, videos, audio files, text). Handling different data types and integrating them for analysis is a core challenge in Big Data.
4. **Veracity**: Refers to the quality and accuracy of data. Because Big Data often comes from various sources, it can be inconsistent, incomplete, or uncertain. Ensuring the reliability and trustworthiness of data is crucial for making accurate decisions based on Big Data analysis.

- **Data Preparation:** Cleaning, transforming, and preprocessing the data to make it suitable for analysis. This includes handling missing values, dealing with outliers, and formatting data.
- **Pattern Discovery:** Employing algorithms to identify patterns, associations, correlations, or anomalies within the data.
- **Predictive Modelling:** Building predictive models that can forecast future trends or outcomes based on historical data.
- **Clustering and Classification:** Grouping similar data points into clusters or categorizing data into predefined classes or categories.
- **Data Visualization:** Presenting the results of data mining in visual formats such as charts, graphs, or reports for easier interpretation.

The Importance of Big Data in Data Mining



Data Preprocessing

- Involves identifying and correcting errors or inconsistencies in a dataset to improve its quality.
- Example: In a customer database, there may be entries with missing values, such as email addresses or phone numbers.
- Data cleaning would involve either filling in these missing values with reasonable estimates or removing records with too many missing values.

Data Integration

- Data integration combines data from different sources into a unified view, providing a comprehensive and coherent dataset.
- Example: An e-commerce company integrates data from various systems, including sales records, inventory databases, and customer information, to create a complete view of its business operations.

Data Transformation

- Data transformation involves converting data into a different format, structure, or scale to make it suitable for analysis or a specific application. Example: Converting temperature readings from Fahrenheit to Celsius is a data transformation. Similarly, aggregating daily sales data into monthly totals is a transformation to a different scale.
- Data reduction aims to decrease the volume but produce the same or similar analytical results. It often involves selecting a representative subset of data. Example: In machine learning, feature selection is a form of data reduction. Choosing the most relevant features (variables) from a dataset while discarding less important ones can improve model performance and reduce computational costs.

EDA

- EDA is the process of summarizing, visualizing, and understanding the main characteristics of a dataset before conducting formal analyses.
- Example: Before building a predictive model, a data scientist might use EDA techniques to create histograms, scatter plots, and summary statistics to uncover trends, anomalies, or relationships in the data.

Data Analysis

1. Univariate analysis focuses on a single variable or feature in a dataset to understand its distribution, central tendencies, and variability. **Example:** If you want to understand the distribution of ages in a population, you can create a histogram or a box plot showing the frequency or density of different age groups.
2. Bivariate analysis involves analyzing the relationship or association between two variables in a dataset. **Example:** To study how the price of a product is affected by its advertising spending, you can create a scatter plot with advertising spending on the x-axis and product price on the y-axis.
3. Multivariate analysis deals with the simultaneous analysis of more than two variables in a dataset to uncover complex relationships and patterns. Example: Principal Component Analysis (PCA) is a multivariate technique that reduces the dimensionality of a dataset while preserving as much variance as possible. It's often used for feature reduction in machine learning.

Data Mining

- Data mining techniques are methods used to extract patterns, knowledge, or valuable information from large datasets.
  Example: Using association rule mining to identify patterns in customer purchase data, such as discovering that customers who buy cereal and milk are also likely to buy bread.

Classification and Prediction Classification

- Classification involves assigning data instances to predefined categories or classes based on their characteristics. For example, classifying emails as spam or not spam based on their content.
- Prediction aims to estimate a numerical or categorical outcome for a data instance based on historical data. For instance, predicting a student's future GPA based on their past academic performance.
- Clustering groups similar data points together based on their inherent characteristics, without predefined categories. Example: Clustering customer data to segment them into distinct groups, such as "high spenders," "budget shoppers," and "window shoppers," based on their shopping behavior.

Association Rule Mining and Anomaly Detection

- Association rule mining identifies patterns or relationships between items in a dataset. It's commonly used in market basket analysis to discover which items are frequently purchased together. Example: Supermarkets analyzing customer purchase data to find associations like "Customers who buy bread also tend to buy butter."
- Anomaly Detection: Anomaly detection identifies data instances that deviate significantly from the norm or exhibit unusual behavior. It's used to find rare events or outliers. Example: Detecting fraudulent credit card transactions by identifying transactions that significantly differ from a user's typical spending behavior.

Machine Learning

Machine learning algorithms are computational techniques that enable computers to learn and make predictions or decisions from data without being explicitly programmed.
Example: Using a decision tree algorithm to predict whether a loan applicant is likely to default based on features like credit score, income, and loan amount.

Supervised learning

- In supervised learning, the algorithm learns from a labelled dataset, where each input data point is associated with a corresponding target or output label.

- The algorithm's goal is to learn a mapping or relationship between the input data and the output labels. It aims to make predictions or classifications based on new, unseen data.
- Example: Consider a spam email classifier. You have a dataset of emails, and each email is labelled as either "spam" or "not spam" (ham). The features of these emails, such as the words used and email metadata, serve as input data.
- In supervised learning, the algorithm learns from this dataset to predict whether incoming emails are spam or not based on their features. It uses the labels (spam or not spam) to train and fine-tune its predictive model.

Unsupervised Learning

- In unsupervised learning, the algorithm works with unlabelled data, meaning it doesn't have access to explicit target labels or categories. Instead, the algorithm tries to discover patterns, structures, or relationships within the data without any prior guidance.
- Unsupervised learning is often used for tasks like clustering, dimensionality reduction, and anomaly detection.
- Example: Imagine you have a dataset of customer purchase history, including what items they bought and when. In unsupervised learning, you might use clustering to group similar customers together based on their purchase behavior. The algorithm would identify patterns within the data, such as customers who frequently buy electronics, customers who prefer clothing, etc., without any predefined categories. This can help businesses understand customer segments for targeted marketing.

Machine Learning Algorithms

- Decision trees are a supervised learning algorithm used for classification and regression tasks. They partition the input data into subsets based on features and recursively make decisions at each node to arrive at a final prediction.
- Random Forests, on the other hand, are an ensemble method that consists of multiple decision trees. They work by aggregating the predictions of multiple decision trees to improve accuracy and reduce overfitting.

Decision vs Random Forest – (Supervised - Require labelled data)

- **Decision trees** are simple to understand and interpret but can be prone to overfitting when they become too deep.
- **Random Forests**, on the other hand, mitigate overfitting by averaging the predictions of many decision trees, making them more robust and accurate. While decision trees can be used for both classification and regression, Random Forests are primarily used for classification tasks.
- **Supervised or Unsupervised** Both Decision Trees and Random Forests are supervised learning algorithms because they require labelled data during training. In the case of classification, they need labelled examples to learn the decision boundaries.

Support Vector Machine

- SVM is a supervised learning algorithm used for classification and regression tasks. It finds a hyperplane or decision boundary that best separates different classes in the input data while maximizing the margin between them.
- SVM can also handle non-linear classification by using kernel functions.
- Difference: SVM aims to find the optimal decision boundary that maximizes the margin between data points of different classes. It is effective in high-dimensional spaces and can handle nonlinear data using kernel tricks.
- SVM tries to find the "best" boundary by maximizing the margin, while other algorithms like decision trees might create simpler boundaries based on recursive splits.
- **Supervised or Unsupervised:** SVM is a supervised learning algorithm, as it relies on labelled data during training to learn the decision boundary.

Neural Networks

- Neural networks are a family of machine learning algorithms inspired by the structure and function of the human brain.
- Deep Learning is a subfield of machine learning that focuses on neural networks with many layers (deep neural networks). Neural networks consist of interconnected nodes or neurons that process and transform data through multiple layers to make predictions.

Challenges of Mining Big Data

- **Volume**: Big data involves massive volumes of information that traditional data mining tools and techniques struggle to handle efficiently.
- **Velocity**: Data streams in at an unprecedented speed, such as social media updates, sensor data, and financial transactions. Realtime or near-real-time processing is essential to extract valuable insights promptly.
- **Veracity**: Big data often contains noisy, incomplete, or inaccurate information. Ensuring data quality and dealing with uncertainties can be a substantial challenge in mining bigdata effectively.
- **Value**: Identifying valuable insights from big data can be challenging. With vast amounts of data, it's easy to get lost in irrelevant information.
- **Privacy and Security**: The sheer amount of data increases the risk of privacy breaches and security threats. Safeguarding sensitive information while allowing datamining is a significant challenge.
- **Scalability**: Traditional data mining algorithms may not scale efficiently to handle big data. Developing scalable algorithms and distributed computing solutions is essential.
- **Interoperability**: Integrating big data tools and platforms into existing IT infrastructures can be complex. Ensuring that new technologies work seamlessly with legacy systems is a challenge.
- **Regulatory Compliance**: Big data mining must adhere to various regulations and legal constraints, such as GDPR in Europe or HIPAA in healthcare. Ensuring compliance while mining data can be challenging.
- **Resource Constraints**: Processing and storing big data require significant computational and storage resources. Ensuring access to these resources can be a challenge, especially for smaller organizations.
- **Ethical Concerns**: As data mining becomes more pervasive, ethical concerns regarding data privacy, surveillance, and potential biases in algorithms need to be addressed.
- **Complexity**: Big data projects often involve complex data preparation, feature engineering, and modelling processes. Managing this complexity and ensuring the interpretability of results can be challenging.

Supervised Machine Learning Implementations

Transformation	Handles Negative Values?	Aggressiveness (Skew Reduction)	How to Use
Log	No	High	np.log(data) (or np.log10(data) for base 10)
Square Root	No	Moderate	np.sqrt(data)
Yeo-Johnson	Yes	Adjustable	df2['BTC'], _ = stats.yeojohnson(df2['BTC'])

Interpreting the ROC

Interpreting the Confusion Matrix

3. Lecture 2 basic Statistics and Data Processing

Importance of Data Exploration and Visualization in Data Science

Data Exploration and Visualization play a fundamental and pivotal role in the field of Data Science, serving as the cornerstone of informed decision-making and insightful analysis. Their importance lies in their ability to unearth hidden patterns, trends, and relationships within complex datasets, translating raw information into actionable insights.

Data Quality Assessment

- Visualization aids in identifying data quality issues such as outliers, missing values, or inconsistencies. These visual cues guide data cleaning and preprocessing, enhancing the accuracy and reliability of subsequent analyses.

Feature Selection and Engineering

- Effective data exploration helps in selecting relevant features for modelling while also inspiring the creation of new features that might enhance predictive performance. This contributes to more robust and accurate machine learning models.

Enhanced Creativity

- Visualization encourages creative thinking and innovative problem-solving. Exploring data visually can lead to the discovery of new angles or perspectives that may not have been evident through traditional numerical analysis alone.

Understanding the Dataset

Contextualizing Analysis: A deep understanding of the dataset provides context to the data scientist. It helps them comprehend the origin, source, and purpose of the data, enabling them to make informed decisions about how to preprocess, analyze, and interpret the information.

Data Quality Assessment: Understanding the dataset allows data scientists to assess the quality and reliability of the data. They can identify issues such as missing values, outliers, duplicates, and inconsistencies, which must be addressed before meaningful analyses can take place.

Feature Selection and Engineering: A thorough understanding of the dataset aids in selecting the most relevant features (variables) for analysis. It also inspires the creation of new features through feature engineering, potentially enhancing the performance of predictive models.

Bias and Limitations: By understanding the dataset's characteristics, data scientists can identify potential biases, limitations, and sources of noise. This awareness is crucial for interpreting results accurately and avoiding erroneous conclusions.

Optimal Analytical Approaches: Different datasets require different analytical techniques. Understanding the dataset helps data scientists choose appropriate statistical methods, machine learning algorithms, and visualization strategies that are best suited to reveal insights from the data.

Domain-Specific Insights: Understanding the dataset in the context of the domain it represents allows data scientists to uncover meaningful insights that might not be apparent through analysis alone. This domain expertise helps interpret the results in a way that aligns

Handling Missing Values, Duplicates, and Outliers:

Missing Values

- When data points are missing, it can lead to skewed analysis and biased results. Handling missing values involves imputing or removing them. For example, in a dataset of survey responses, some participants might not have provided their age. You can impute missing ages with the median age of the respondents.

Duplicates

- Duplicate entries can distort analysis and lead to overrepresentation. Detecting and removing duplicates ensures data integrity. For instance, in an e-commerce dataset, there might be

multiple identical orders due to system errors. Removing duplicates ensures accurate order counts.

Outliers

- Outliers are data points that significantly deviate from the norm. Outliers can affect statistical measures and model performance. Addressing outliers may involve removing or transforming them. In a dataset of exam scores, a single exceptionally high score might be an outlier. You can replace it with a more reasonable value based on the distribution of scores.

Data Transformation and Normalization Techniques

Data Transformation

- Data transformation involves converting data into a more suitable format or scale. For instance, transforming skewed data using logarithms can help achieve a more normal distribution. In a dataset of income levels, applying a logarithmic transformation can reduce the impact of extreme incomes.

Normalization

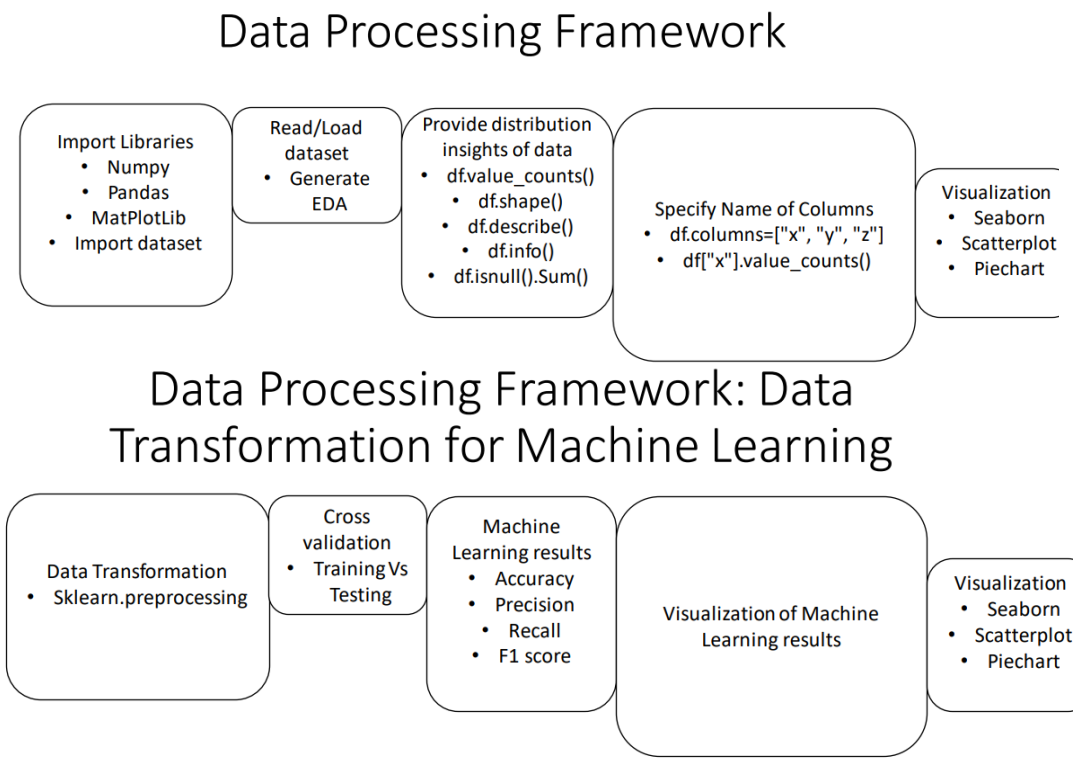
- Normalization scales data to a common range, making comparisons meaningful. In machine learning algorithms, normalized data prevents certain features from dominating others. For example, consider a dataset with attributes like age (0-100) and income (0-100000). Normalizing these features to a 0-1 range ensures balanced contributions to the analysis.

Standardization

- Standardization scales data to have zero mean and unit variance. This is particularly useful for algorithms that assume normally distributed data. For instance, in a dataset containing height and weight, standardization ensures that both attributes contribute equally to clustering algorithms.

Encoding Categorical Variables

- Machine learning models often require numerical inputs. Categorical variables s (e.g., "red," "blue," "green") need to be encoded into numerical values.



DataFrame

is a two-dimensional, size-mutable, and potentially heterogeneous tabular data structure in the panda's library of Python. It is similar to a table in a database or an Excel spreadsheet, where data is arranged in rows and columns.

Exploratory data analysis (EDA)

Thes packages in Python is extremely beneficial for gaining insights into the dataset, cleaning and preparing the data, and guiding the feature engineering and modelling process. These packages provide a range of tools for summarizing, visualizing, and understanding the data, making them essential for any data science. One library that you can use for this is Sweetviz. The code for Sweetviz is just down below.

```
1. import sweetviz as sv report = sv.analyze(UGRansome)
2.  report.show_html('report.html')
```

Central Tendency

Consider the dataset: [1, 2, 2, 3, 4, 7, 9]

- **Mean**:

$$\mu = \frac{1+2+2+3+4+7+9}{7} = \frac{28}{7} = 4$$

- **Median**: Since there are 7 values (an odd number), the median is the 4th value when the data is ordered, which is 3.

- **Standard Deviation**:

$$\sigma = \sqrt{\frac{1}{7}((1-4)^2 + (2-4)^2 + (2-4)^2 + (3-4)^2 + (4-4)^2 + (7-4)^2 + (9-4)^2}$$

$$\sigma = \sqrt{\frac{1}{7}(9+4+4+1+0+9+25)} = \sqrt{\frac{52}{7}} \approx 2.72$$

In summary:

- The **mean** is the average value.
- The **median** is the middle value in an ordered dataset.
- The **standard deviation** measures the spread of the data around the mean.

These metrics—mean, median, and standard deviation—provide important insights into the characteristics of a dataset:

Mean

Central Tendency: The mean gives you an idea of the average value in the dataset.
Overall Level: It indicates the overall level of the values in the dataset.
Effect of Outliers: Since the mean is sensitive to outliers, a few extremely high or low values can significantly affect it. This can be useful to know if the dataset contains outliers.

Median

Central Tendency: Like the mean, the median gives you an idea of the central tendency of the dataset.
Middle Value: The median is the middle value when the data is ordered, so it represents the 50th percentile.
Resilience to Outliers: The median is not affected by outliers, making it a better measure of central tendency when the dataset contains extreme values.

Standard Deviation

Spread: The standard deviation tells you how spread out the values in the dataset are around the mean.
Variability: A low standard deviation means the values are close to the mean, indicating low variability. A high standard deviation means the values are spread out over a wider range, indicating high variability.
Data Consistency: It helps to understand the consistency of the data. For example, in quality control processes, a high standard deviation might indicate inconsistent product quality.

What These Metrics Tell You About a Dataset

The mean and median provide information about the central tendency or typical value of the data.

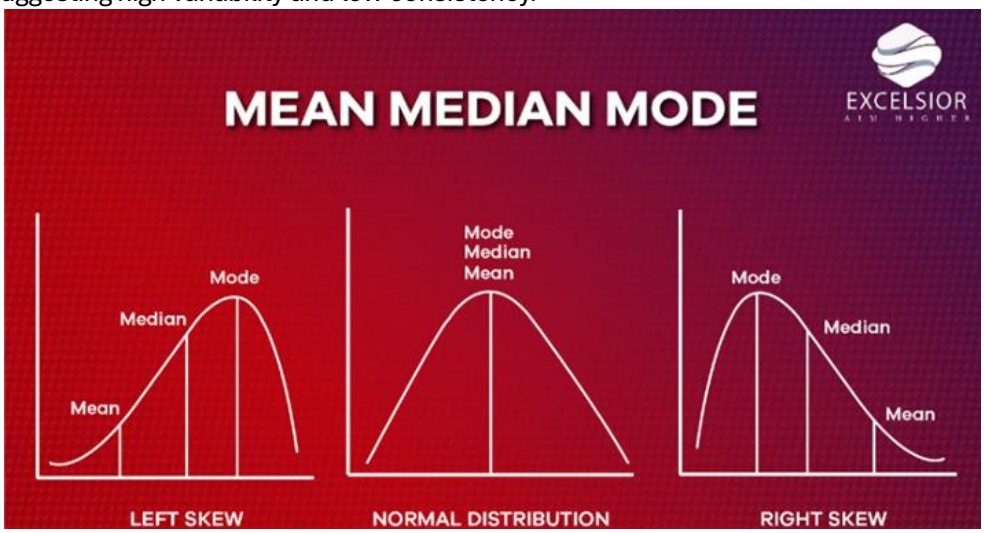
- If the mean and median are close, the dataset is likely symmetrically distributed.
- If the mean is significantly higher or lower than the median, the dataset may be skewed.

Skewness

- If the mean is greater than the median, the data might be right-skewed (positively skewed).
- If the mean is less than the median, the data might be left-skewed (negatively skewed).

Variability and Consistency

- A low standard deviation indicates that the data points are close to the mean, suggesting low variability and high consistency.
- A high standard deviation indicates that the data points are spread out over a larger range, suggesting high variability and low consistency.



Examples

Income Data

- In income data, the mean might be higher than the median if there are a few individuals with very high incomes (right-skewed distribution). The median might give a better sense of a "typical" income.

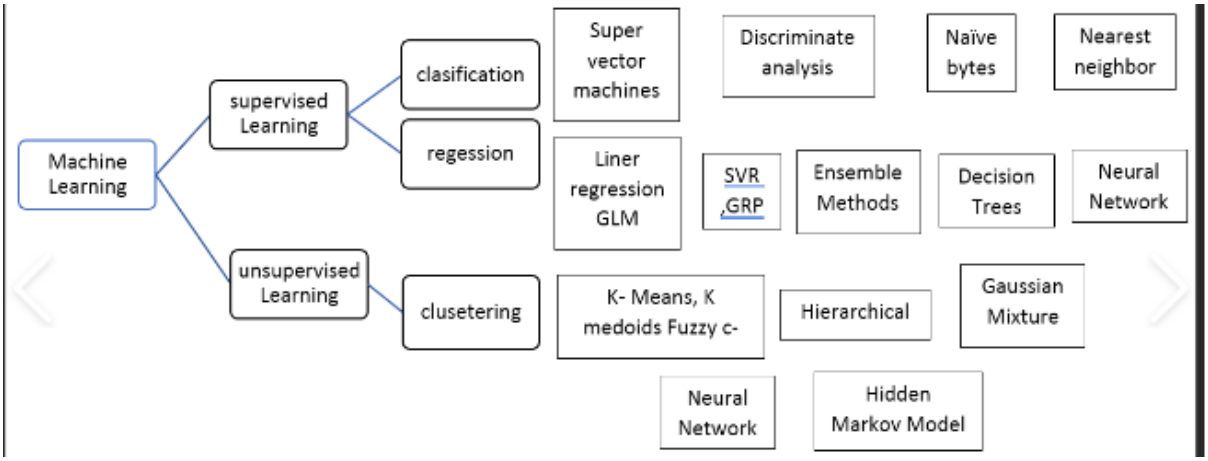
Test Scores

- For test scores of a class, if the standard deviation is low, it indicates that most students scored similarly. A high standard deviation would indicate a wide range of scores, with students scoring very differently from each other.

Quality Control

- In manufacturing, a low standard deviation of product dimensions might indicate high precision in the manufacturing process, whereas a high standard deviation might indicate issues with the process consistency.

4. Lecture 3 (Supervised and Ensemble Learning)



Naive Bayes

- Is a probabilistic classifier based on Bayes' theorem with an assumption of independence between features. It is often used for classification tasks and works particularly well for large datasets.

Bayes' Theorem

Naive Bayes uses Bayes' theorem to calculate the probability of a class given the features.

$$P(C \mid X) = \frac{P(X \mid C) \cdot P(C)}{P(X)}$$

where C is the class, and X represents the features.

Naive Assumption

The "naive" part refers to the assumption that all features are independent given the class. This simplifies the computation of P (X | C)$^{p(X|C)}$ to

$$P(X \mid C) = \prod_{i=1}^{n} P(x_i \mid C)$$

- where xi are the individual features.

Types of Naive Bayes classifiers based on the type of features

1. Gaussian Naive Bayes - Assumes that features follow a Gaussian distribution.
2. Multinomial Naive Bayes - Used for discrete features like word counts in text classification.
3. Bernoulli Naive Bayes - Used for binary/boolean features.

Strengths

- Simple and easy to implement.

- Works well with high-dimensional data.
- Fast and Scalable Naive Bayes is computationally efficient, requiring linear time for training and prediction. This makes it suitable for large datasets.
- Handles Missing Data Can handle missing values naturally by ignoring the feature during probability computation.
- Effective for Text Classification Particularly well-suited for text classification tasks like spam detection, sentiment analysis, and document categorization due to the bag-of-words model often used in these applications.
- Performs Well with Small Data Even with small datasets, Naive Bayes can produce reasonably good results compared to more complex models.
- Probabilistic Output Provides class probabilities, which can be useful for decision-making processes requiring uncertainty estimation.
- Few Hyperparameters Minimal tuning is needed, making it easy to use and interpret.

Weaknesses

- The assumption of feature independence is often unrealistic, which can affect performance.
- Sensitivity to Data Imbalance Performs poorly when the dataset is imbalanced since it assumes equal prior probabilities unless explicitly specified.
- Limited Expressiveness Due to its simplistic assumption of independence, it may fail to capture complex relationships in data.
- Zero-Frequency Problem If a category in a feature was not seen in the training set, the probability for that category becomes zero. This can be addressed using smoothing techniques like Laplace smoothing.
- Poor Performance on Continuous Data Without Proper Preprocessing Naive Bayes works better with categorical or discrete data. For continuous data, assumptions about the distribution (e.g., Gaussian) need to hold true, which may not always be the case.
- Not Ideal for Feature Interactions Cannot model interdependencies or correlations between features effectively. More sophisticated models like Random Forests or Support Vector Machines can outperform Naive Bayes when feature relationships are critical.
- Over-reliance on Feature Relevance Features with no significant correlation to the target variable may still influence the decision-making process, especially when there is noise.

Support Vector Machine (SVM)

- Support Vector Machine (SVM) is a supervised learning algorithm used for classification and regression tasks. It finds the hyperplane that best separates different classes in the feature space.

Hyperplane

- SVM tries to find the optimal hyperplane that maximizes the margin between two classes. The margin is the distance between the hyperplane and the nearest data points from each class, which are called support vectors.

Kernel Trick

- SVM can be extended to handle non-linear relationships using the kernel trick, which transforms the data into a higher-dimensional space where a linear hyperplane can be used to separate the classes.

Regularization

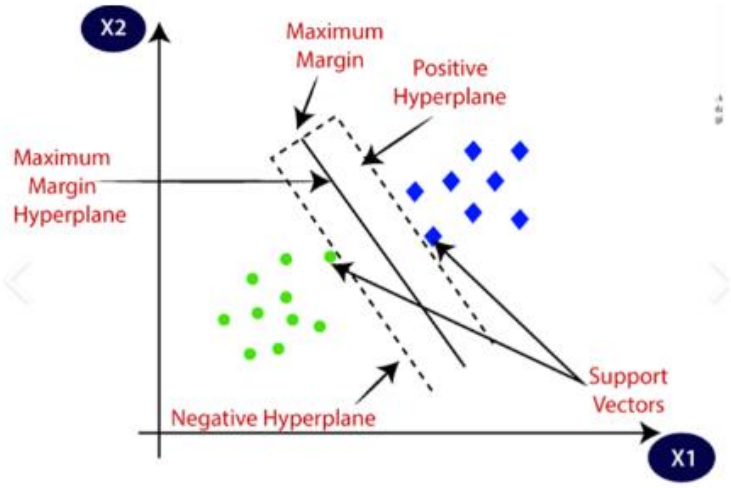
- SVM includes a regularization parameter that controls the trade-off between maximizing the margin and minimizing classification error.

Strengths:

- Effective in high-dimensional spaces.
- Can handle non-linear boundaries using kernels.

Weaknesses:

- Can be computationally expensive for large datasets.
- Requires careful tuning of hyperparameters.



Random Forest

- Random Forest is an ensemble learning method that combines multiple decision trees to improve classification and regression performance. It uses the following principles:

Bootstrap Aggregating (Bagging)

- Random forests build multiple decision trees using different subsets of the training data. Each tree is trained on a random sample of the data with replacement (bootstrap sample).

Random Feature Selection

- When splitting nodes in a tree, random forests consider a random subset of features. This reduces correlation between trees and improves generalization.

Voting/Averaging

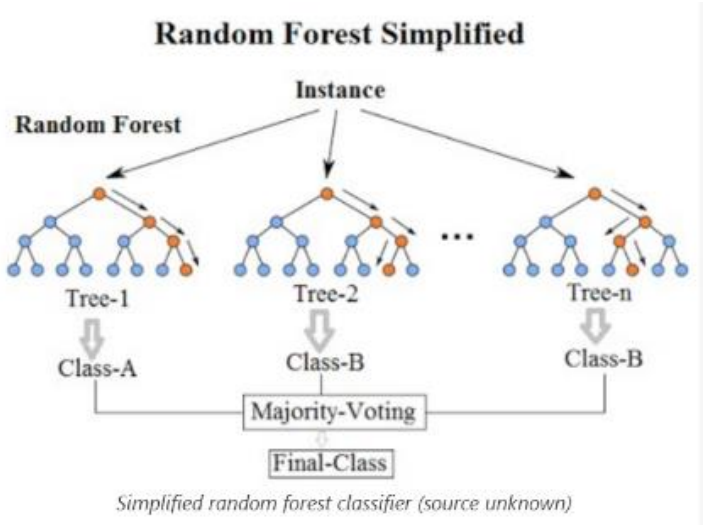
- For classification, the final prediction is made by majority voting from all decision trees. For regression, it is the average of predictions from all trees.

Strengths

- Handles both classification and regression tasks.
- Reduces overfitting compared to individual decision trees.
- Can handle large datasets with many features.

Weaknesses

- Can be computationally intensive and slower to predict than individual decision trees.
- Less interpretable than a single decision tree.



Simplified random forest classifier (source unknown)

Ensemble Learning

- Ensemble Learning is a machine learning paradigm that combines multiple models to produce a better overall performance than individual models.
- The idea is to leverage the strengths of different models and mitigate their weaknesses.

Strengths

- Often improves accuracy and robustness compared to individual models.
- Can handle a wide variety of data and problems.

Weaknesses

- Can be complex to implement and tune.
- May require more computational resources.

Types of Ensemble Methods

Bagging (Bootstrap Aggregating)

Builds multiple models (e.g., decision trees) using different subsets of the data. The final prediction is an aggregation of predictions from all models. Random Forest is a popular bagging method.

Boosting

Sequentially builds models, where each new model corrects errors made by the previous ones. Examples include AdaBoost and Gradient Boosting Machines (GBM).

Stacking

Combines multiple models (base learners) and uses another model (meta-learner) to aggregate their predictions. The base learners might use different algorithms, and the meta-learner combines their outputs.

Summary

- Naive Bayes is a probabilistic classifier based on independence assumptions.
- SVM is a powerful classifier that finds optimal decision boundaries.
- Random Forest is an ensemble of decision trees that improves performance through averaging and randomization.
- Ensemble Learning combines multiple models to improve performance and generalization.

5. Lecture 4 : Applied Statistical Analysis

- Applied statistical analysis refers to the use of statistical methods and techniques to analyze real-world data in order to make informed decisions, draw conclusions, and solve practical problems. It involves the application of statistical principles to various fields such as science, engineering, business, healthcare, social sciences, and more. The primary goal of applied statistical analysis is to extract meaningful information from data, uncover patterns and trends, test hypotheses, and make predictions or recommendations based on the data.
- Applied statistics is the root of data analysis. The practice of applied statistics involves analyzing data to help define and determine business needs. Modern workplaces are overwhelmed with big data and are looking for statisticians, data analysts, data scientists, and other professionals with applied statistics knowledge who can organize, analyze, and use data to solve real-world problems.

What are key aspects of Applied Statistical Analysis?

1. Data Collection

Gather Data (continued…)

diabetes_dataset.csv

diabetes_dataset.csv | Diabetes_study.ipynb

Delimiter: ,

	AGE	SEX	BMI	BP	S1	S2	S3	S4	S5
1	59	2	32.1	101	157	93.2	38	4	4.8598
2	48	1	21.6	87	183	103.2	70	3	3.8918
3	72	2	30.5	93	156	93.6	41	4	4.6728
4	24	1	25.3	84	198	131.4	40	5	4.8903
5	50	1	23	101	192	125.4	52	4	4.2905
6	23	1	22.6	89	139	64.8	61	2	4.1897
7	36	2	22	90	160	99.6	50	3	3.9512
8	66	2	26.2	114	255	185	56	4.55	4.2485
9	60	2	32.1	83	179	119.4	42	4	4.4773
10	29	1	30	85	180	93.4	43	4	5.3845
11	22	1	18.6	97	114	57.6	46	2	3.9512
12	56	2	28	85	184	144.8	32	6	3.5835
13	53	1	23.7	92	186	109.2	62	3	4.3041

```
import pandas as pd
data = pd.read_csv('diabetes_dataset.csv')
```

2. Data Cleaning and Preprocessing

Data points and features

```
type(data)
```

```
pandas.core.frame.DataFrame
```

```
data.shape
```

```
(442, 11)
```

10 Features in order:

- AGE age in years
- SEX sex
- BMI body mass index
- BP average blood pressure
- S1 tc, total serum cholesterol
- S2 ldl, low-density lipoproteins
- S3 hdl, high-density lipoproteins
- S4 tch, total cholesterol / HDL
- S5 ltg, possibly log of serum triglycerides level
- S6 glu, blood sugar level

Target variable:

- Y response

Cleaning data

```
pd.isnull(data).any()
```

```
AGE     False
SEX     False
BMI     False
BP      False
S1      False
S2      False
S3      False
S4      False
S5      False
S6      False
Y       False
dtype: bool
```

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 442 entries, 0 to 441
Data columns (total 11 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   AGE     442 non-null    int64
 1   SEX     442 non-null    int64
 2   BMI     442 non-null    float64
 3   BP      442 non-null    float64
 4   S1      442 non-null    int64
 5   S2      442 non-null    float64
 6   S3      442 non-null    float64
 7   S4      442 non-null    float64
 8   S5      442 non-null    float64
 9   S6      442 non-null    int64
 10  Y       442 non-null    int64
dtypes: float64(6), int64(5)
memory usage: 38.1 KB
```

3. Descriptive Statistics

Descriptive Statistics

```
data['Y'].min()
```

```
25
```

```
data['Y'].max()
```

```
346
```

```
data.min()
```

```
AGE     19.0000
SEX      1.0000
BMI     18.0000
BP      62.0000
S1      97.0000
S2      41.6000
S3      22.0000
S4       2.0000
S5       3.2581
S6      58.0000
Y       25.0000
dtype: float64
```

```
data.max()
```

```
AGE     79.000
SEX      2.000
BMI     42.200
BP     133.000
S1     301.000
S2     242.400
S3      99.000
S4       9.090
S5       6.107
S6     124.000
Y      346.000
dtype: float64
```

```
data.head()
```

	AGE	SEX	BMI	BP	S1	S2	S3	S4	S5	S6	Y
0	59	2	32.1	101.0	157	93.2	38.0	4.0	4.8598	87	151
1	48	1	21.6	87.0	183	103.2	70.0	3.0	3.8918	69	75
2	72	2	30.5	93.0	156	93.6	41.0	4.0	4.6728	85	141
3	24	1	25.3	84.0	198	131.4	40.0	5.0	4.8903	89	206
4	50	1	23.0	101.0	192	125.4	52.0	4.0	4.2905	80	135

```
data.tail()
```

	AGE	SEX	BMI	BP	S1	S2	S3	S4	S5	S6	Y
437	60	2	28.2	112.00	185	113.8	42.0	4.00	4.9836	93	178
438	47	2	24.9	75.00	225	166.0	42.0	5.00	4.4427	102	104
439	60	2	24.9	99.67	162	106.6	43.0	3.77	4.1271	95	132
440	36	1	30.0	95.00	201	125.2	42.0	4.79	5.1299	85	220
441	36	1	19.6	71.00	250	133.2	97.0	3.00	4.5951	92	57

Descriptive Statistics (continued…)

```
data.describe()
```

	AGE	SEX	BMI	BP	S1	S2	S3	S4	S5	S6	Y
count	442.000000	442.000000	442.000000	442.000000	442.000000	442.000000	442.000000	442.000000	442.000000	442.000000	442.000000
mean	48.518100	1.468326	26.375792	94.647014	189.140271	115.439140	49.788462	4.070249	4.641411	91.260181	152.133484
std	13.109028	0.499561	4.418122	13.831283	34.608052	30.413081	12.934202	1.290450	0.522391	11.496335	77.093005
min	19.000000	1.000000	18.000000	62.000000	97.000000	41.600000	22.000000	2.000000	3.258100	58.000000	25.000000
25%	38.250000	1.000000	23.200000	84.000000	164.250000	96.050000	40.250000	3.000000	4.276700	83.250000	87.000000
50%	50.000000	1.000000	25.700000	93.000000	186.000000	113.000000	48.000000	4.000000	4.620050	91.000000	140.500000
75%	59.000000	2.000000	29.275000	105.000000	209.750000	134.500000	57.750000	5.000000	4.997200	98.000000	211.500000
max	79.000000	2.000000	42.200000	133.000000	301.000000	242.400000	99.000000	9.090000	6.107000	124.000000	346.000000

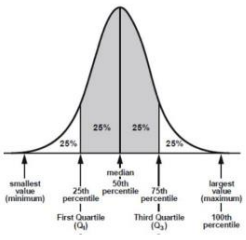
```
data.mean()
```

```
AGE     48.518100
SEX      1.468326
BMI     26.375792
BP      94.647014
S1     189.140271
S2     115.439140
S3      49.788462
S4       4.070249
S5       4.641411
S6      91.260181
Y      152.133484
dtype: float64
```

```
data.median()
```

```
AGE     50.00000
SEX      1.00000
BMI     25.70000
BP      93.00000
S1     186.00000
S2     113.00000
S3      48.00000
S4       4.00000
S5       4.62005
S6      91.00000
Y      140.50000
dtype: float64
```

- Mean: The average
- Median: The midpoint of the distribution. I.e., the middle value



4. Exploratory Data Analysis

Data exploration

```
data.head()
```

	AGE	SEX	BMI	BP	S1	S2	S3	S4	S5	S6	Y
0	59	2	32.1	101.0	157	93.2	38.0	4.0	4.8598	87	151
1	48	1	21.6	87.0	183	103.2	70.0	3.0	3.8918	69	75
2	72	2	30.5	93.0	156	93.6	41.0	4.0	4.6728	85	141
3	24	1	25.3	84.0	198	131.4	40.0	5.0	4.8903	89	206
4	50	1	23.0	101.0	192	125.4	52.0	4.0	4.2905	80	135

```
data.tail()
```

	AGE	SEX	BMI	BP	S1	S2	S3	S4	S5	S6	Y
437	60	2	28.2	112.00	185	113.8	42.0	4.00	4.9836	93	178
438	47	2	24.9	75.00	225	166.0	42.0	5.00	4.4427	102	104
439	60	2	24.9	99.67	162	106.6	43.0	3.77	4.1271	95	132
440	36	1	30.0	95.00	201	125.2	42.0	4.79	5.1299	85	220
441	36	1	19.6	71.00	250	133.2	97.0	3.00	4.5951	92	57

```
data.count()
```

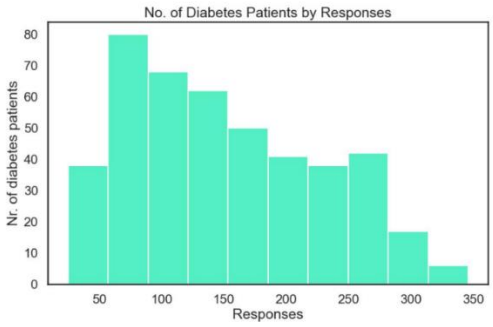
```
AGE     442
SEX     442
BMI     442
BP      442
S1      442
S2      442
S3      442
S4      442
S5      442
S6      442
Y       442
dtype: int64
```

- Data visualization also help us make sense of the data at the Exploration Stage

5. Inferential Statistics
6. Regression Analysis
7. Data Visualization

Visualising Data

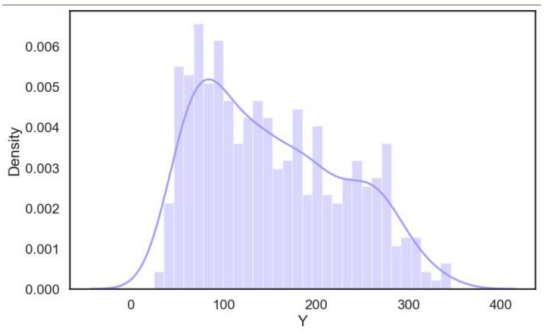
```
plt.figure(figsize=(10, 6))
plt.hist(data['Y'], ec='#ffffff', color='#55efc4')
plt.xlabel('Responses')
plt.ylabel('Nr. of diabetes patients')
plt.title('No. of Diabetes Patients by Responses')
plt.show()
```

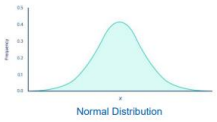


- Data visualization also help us make sense of the data at the Exploration Stage
- Used for data distribution & outliers
- Useful for e.g., identifying Normal vs Skewed Distributions

Visualising Data (continued…)

```
plt.figure(figsize=(10, 6))
sns.distplot(data['Y'], bins=30, color='#a29bfe')
plt.show()
```





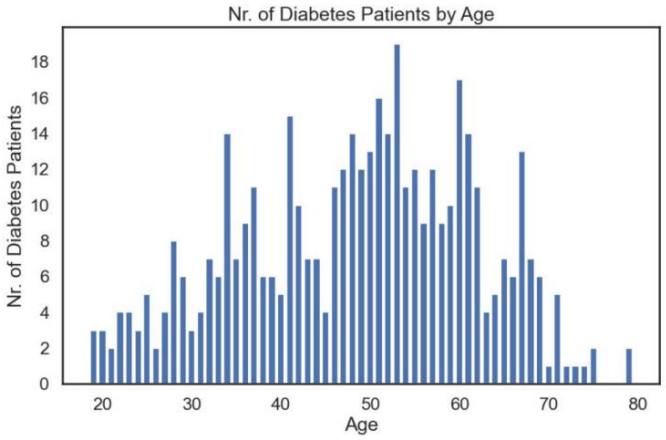
Visualising Data (continued…)

```
data['AGE'].mean()
```

```
48.51809954751131
```

```
data['AGE'].median()
```

```
50.0
```




```
frequency = data['AGE'].value_counts()

plt.figure(figsize=(10, 6))
plt.xlabel('Age')
plt.ylabel('Nr. of Diabetes Patients')
custom_y_ticks = [0, 2, 4, 6, 8, 10, 12, 14, 16, 18, 20]
plt.yticks(custom_y_ticks)
plt.title('Nr. of Diabetes Patients by Age')
plt.bar(frequency.index, height=frequency)
plt.show()
```

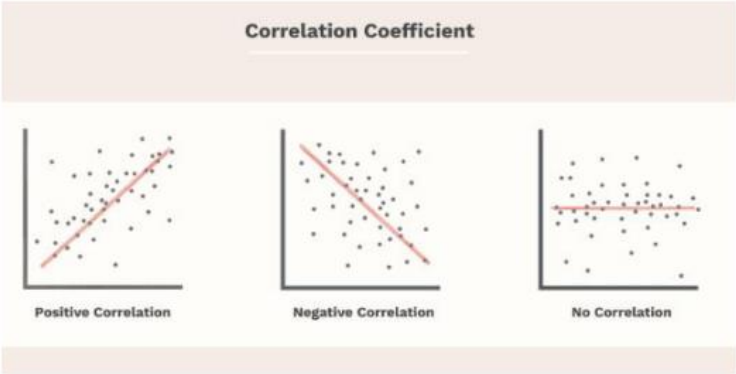

8. Interpretation and Reporting

Correlation

Correlation is a statistical measure that describes the extent to which two variables change together (i.e., their relationship).

- 1 is a perfect positive correlation.
- 0 means there is no correlation.
- -1 is a perfect negative correlation.

Correlation (a linear relationship): is important because we want to include features that have the right strength and direction (i.e., features that are correlated with the target)



Correlation

```
data['Y'].corr(data['BMI'])
```

```
0.5864501344746885
```

```
data['BMI'].corr(data['AGE'])
```

```
0.18508466614655544
```

10 Features in order:

- AGE age in years
- SEX sex
- BMI body mass index
- BP average blood pressure
- S1 tc, total serum cholesterol
- S2 ldl, low-density lipoproteins
- S3 hdl, high-density lipoproteins
- S4 tch, total cholesterol / HDL
- S5 ltg, possibly log of serum triglycerides level
- S6 glu, blood sugar level

Target variable:

- Y response

```
data.corr()
```

	AGE	SEX	BMI	BP	S1	S2	S3	S4	S5	S6	Y
AGE	1.000000	0.173737	0.185085	0.335428	0.260061	0.219243	-0.075181	0.203841	0.270774	0.301731	0.187889
SEX	0.173737	1.000000	0.088161	0.241010	0.035277	0.142637	-0.379090	0.332115	0.149916	0.208133	0.043062
BMI	0.185085	0.088161	1.000000	0.395411	0.249777	0.261170	-0.366811	0.413807	0.446157	0.388680	0.586450
BP	0.335428	0.241010	0.395411	1.000000	0.242464	0.185548	-0.178762	0.257650	0.393480	0.390430	0.441482
S1	0.260061	0.035277	0.249777	0.242464	1.000000	0.896663	0.051519	0.542207	0.515503	0.325717	0.212022
S2	0.219243	0.142637	0.261170	0.185548	0.896663	1.000000	-0.196455	0.659817	0.318357	0.290600	0.174054
S3	-0.075181	-0.379090	-0.366811	-0.178762	0.051519	-0.196455	1.000000	-0.738493	-0.398577	-0.273697	-0.394789
S4	0.203841	0.332115	0.413807	0.257650	0.542207	0.659817	-0.738493	1.000000	0.617859	0.417212	0.430453
S5	0.270774	0.149916	0.446157	0.393480	0.515503	0.318357	-0.398577	0.617859	1.000000	0.464669	0.565883
S6	0.301731	0.208133	0.388680	0.390430	0.325717	0.290600	-0.273697	0.417212	0.464669	1.000000	0.382483
Y	0.187889	0.043062	0.586450	0.441482	0.212022	0.174054	-0.394789	0.430453	0.565883	0.382483	1.000000

UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

Faculty of Engineering, Built Environment and Information Technology
Fakulteit Ingenieurswese, Bou-omgewing en Inligtingtegnologie / Lefapha la Boetšenere, Tikologo ya Kago le Theknolotši ya Tshedimošo

Correlation (continued…)

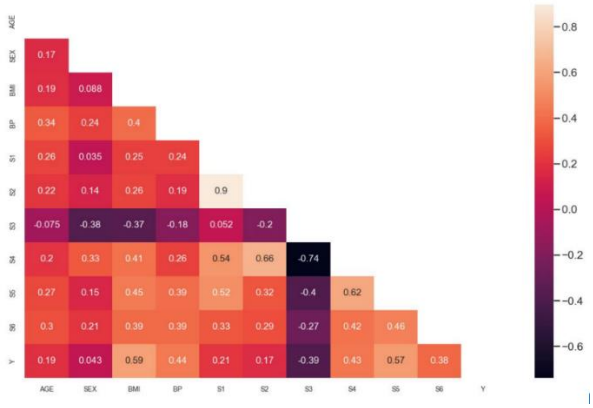
```
mask = np.zeros_like(data.corr())
triangle_indices = np.triu_indices_from(mask)
mask[triangle_indices] = True
mask
```

```
array([[1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.],
       [0., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.],
       [0., 0., 1., 1., 1., 1., 1., 1., 1., 1., 1.],
       [0., 0., 0., 1., 1., 1., 1., 1., 1., 1., 1.],
       [0., 0., 0., 0., 1., 1., 1., 1., 1., 1., 1.],
       [0., 0., 0., 0., 0., 1., 1., 1., 1., 1., 1.],
       [0., 0., 0., 0., 0., 0., 1., 1., 1., 1., 1.],
       [0., 0., 0., 0., 0., 0., 0., 1., 1., 1., 1.],
       [0., 0., 0., 0., 0., 0., 0., 0., 1., 1., 1.],
       [0., 0., 0., 0., 0., 0., 0., 0., 0., 1., 1.],
       [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 1.]])
```

Correlation (continued…)

```
plt.figure(figsize=(16, 10))
sns.heatmap(data.corr(), mask=mask, annot=True, annot_kws={'size': 14})
plt.xticks(fontsize=10)
plt.yticks(fontsize=10)
plt.show()
```

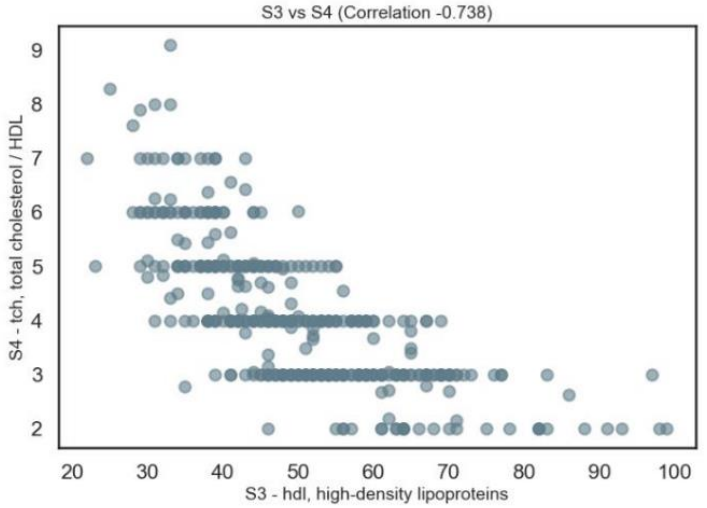
Correlation (continued…)



```python
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
# df2 is your dataframe
plt.figure(figsize=(17, 6))
corr = df2.corr(method='spearman')
my_m = np.triu(corr)
sns.heatmap(corr, mask=my_m, annot=True, cmap="Set2")
plt.show()
correlation_matrix = df2.corr()
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm')
plt.show()
```

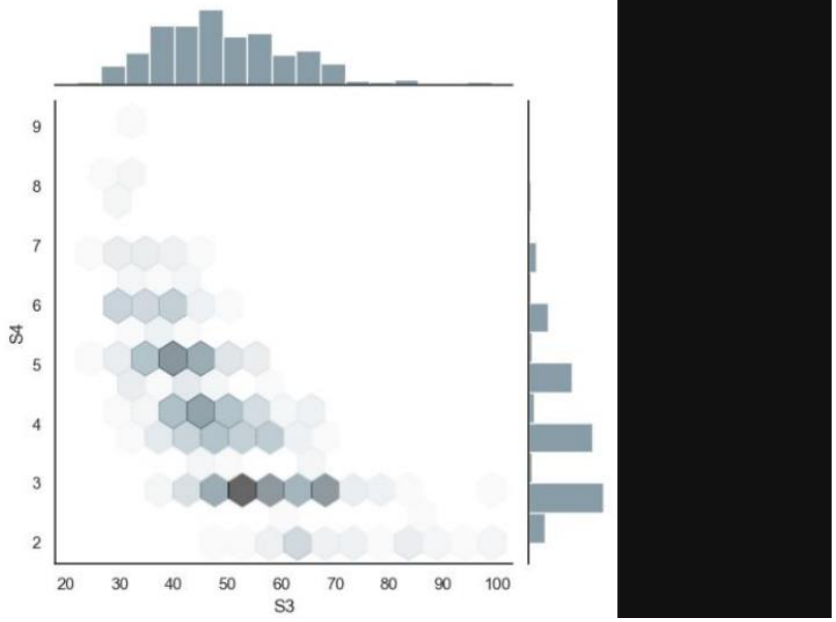
Correlation (continued…)

```python
# Scatter plot between S4 and S3 Correlation
s3_s4_corr = round(data['S3'].corr(data['S4']), 3)
plt.figure(figsize=(9, 6))
plt.scatter(x=data['S3'], y=data['S4'], alpha=0.6, s=80, color='#607D8B')
plt.title(f'S3 vs S4 (Correlation {s3_s4_corr})', fontsize=14)
plt.xlabel('S3 - hdl, high-density lipoproteins', fontsize=14)
plt.ylabel('S4 - tch, total cholesterol / HDL', fontsize=14)
plt.show()
```



Correlation (continued…)

```
sns.set()
sns.set_context('notebook')
sns.set_style('white')
sns.jointplot(x=data['S3'], y=data['S4'], height=6, color='#607D8B', kind='hex', joint_kws={'alpha': 0.6})
plt.show()
```



Machine Learning: Training Split

Training & Test Dataset Split

```
responses = data['Y']
features = data.drop('Y', axis=1)

X_train, X_test, y_train, y_test = train_test_split(features, responses,
                                                    test_size=0.2, random_state=30)


len(X_train)/len(features)
```

```
len(X_test)/len(features)
```

```
0.20135746606334842
```

Multivariable Linear Regression

Multivariable Linear Regression

```
regr = LinearRegression()
regr.fit(X_train, y_train)

print('Training data r-squared:', regr.score(X_train, y_train))
print('Test data r-squared:', regr.score(X_test, y_test))

print('Intercept', regr.intercept_)
pd.DataFrame(data=regr.coef_, index=X_train.columns, columns=['coef'])
```

- Linear regression helps us understand how changes in one or more variables are associated with changes in another variable.

Multivariable Linear Regression (continued…)

```
Training data r-squared: 0.5264783626678893
Test data r-squared: 0.4653044632644133
Intercept -368.2185304349134
```

	coef
AGE	0.029063
SEX	-23.059093
BMI	5.602677
BP	1.254404
S1	-1.334738
S2	0.955752
S3	0.588029
S4	3.158733
S5	78.918445
S6	0.226053

Data Transformation

Data Transformation

```
data['Y'].skew()
```

```
0.44056293407014124
```

```
y_log = np.log(data['Y'])
y_log.tail()
```

```
437    5.181784
438    4.644391
439    4.882802
440    5.393628
441    4.043051
Name: Y, dtype: float64
```

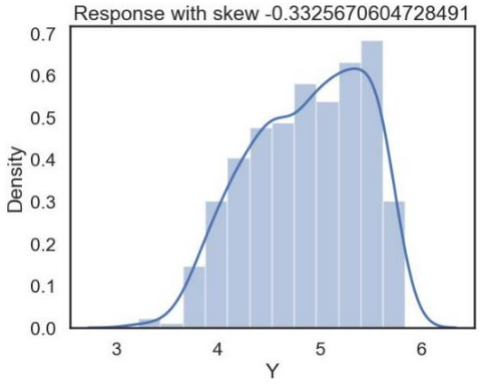
```
y_log.skew()
```

```
-0.3325670604728491
```

- Logarithms (log) helps transform the data before you fit the linear regression line - useful if the data is skew

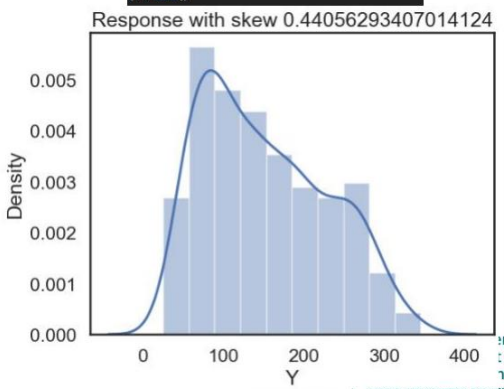
Data Transformation (continued…)

```
sns.distplot(y_log)
plt.title(f'Response with skew {y_log.skew()}')
plt.show()
```

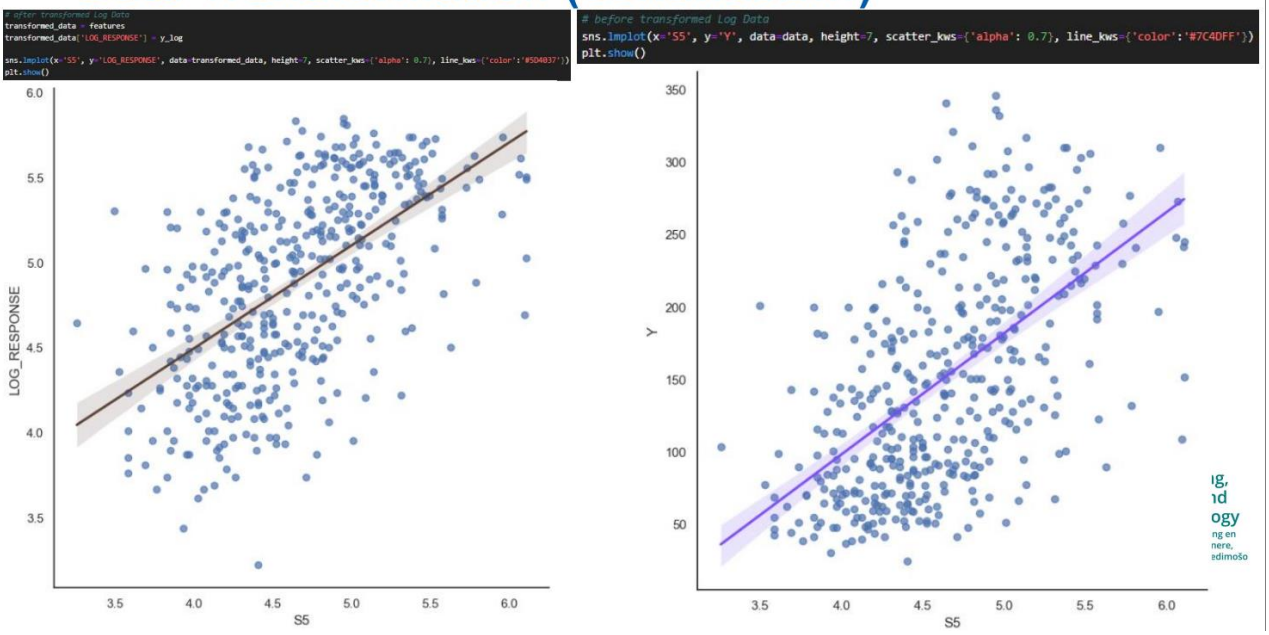



```
tmpdata = data['Y']

sns.distplot(tmpdata)
plt.title(f'Response with skew {tmpdata.skew()}')
plt.show()
```



Data Transformation (continued…)



P Values and Evaluating Coefficients

P values & Evaluating Coefficients

```
X_incl_const = sm.add_constant(X_train)

model = sm.OLS(y_train, X_incl_const)
results = model.fit()

pd.DataFrame({'coef': results.params, 'p-value': round(results.pvalues, 3)})
```

	coef	p-value
const	-368.218530	0.000
AGE	0.029063	0.905
SEX	-23.059093	0.000
BMI	5.602677	0.000
BP	1.254404	0.000
S1	-1.334738	0.031
S2	0.955752	0.096
S3	0.588029	0.487
S4	3.158733	0.624
S5	78.918445	0.000
S6	0.226053	0.457

- P ≤ **0,05** is considered statistically significant.

Faculty of Engineering, Built Environment and Information Technology

Make today matter

6. Lecture 4 (Unsupervised Learning and Deep Learning)

ROC - ROC stands for Receiver Operating Characteristic

ROC stands for Receiver Operating Characteristic. It's a graphical plot that illustrates the diagnostic ability of a binary classifier system as its discrimination threshold is varied.:

- True Positive Rate (TPR) or Sensitivity: This is plotted on the y-axis. It represents the proportion of actual positives correctly identified by the model (e.g., the percentage of crypto jacking cases correctly detected).
- False Positive Rate (FPR): This is plotted on the x-axis. It represents the proportion of actual negatives incorrectly identified as positives (e.g., the percentage of normal cases incorrectly flagged as crypto jacking).

The ROC curve shows the trade-off between sensitivity and specificity (1 - FPR). A model with a curve closer to the top-left corner indicates a better performance, as it shows a higher true positive rate with a lower false positive rate.

The area under the ROC curve (AUC) is often used as a single metric to summarize the model's performance.

- An AUC of 1.0 represents a perfect model, while an AUC of 0.5 suggests no discrimination, akin to random guessing.

How to Interpret the Curve

Curve Shape

The closer the curve is to the **top-left corner**, the better the model's performance.
High TPR (Sensitivity).
Low FPR (few false positives).

Trade-offs

- Moving along the curve corresponds to changing the classification threshold.
- **Higher thresholds**: Fewer positive predictions → Higher Precision, Lower Recall.
- **Lower thresholds**: More positive predictions → Higher Recall, Lower Precision.

Optimal Threshold:

- Choose a threshold where TPR is high, and FPR is low, depending on the problem context.
- If misclassifying positives (False Negatives) is critical (e.g., in medical diagnosis), aim for a **high TPR**.
- If misclassifying negatives (False Positives) is costly (e.g., in fraud detection), aim for a **low FPR**.

AUC Interpretation

- **AUC = 1.0**: Perfect model (predicts all positives and negatives correctly).
- **AUC = 0.9 - 1.0**: Excellent model.
- **AUC = 0.8 - 0.9**: Good model.
- **AUC = 0.7 - 0.8**: Fair model.
- **AUC = 0.5 - 0.7**: Poor model (better than random but needs improvement).
- **AUC = 0.5**: No discrimination (random guessing).

Unsupervised learning for image processing

- Unsupervised learning for image processing involves analyzing and extracting patterns from images without using labelled data. Unlike supervised learning, where models are trained on labelled datasets (i.e., images with corresponding labels), unsupervised learning works with unlabelled data, discovering hidden structures and relationships in the data.

Common techniques and their applications in image processing

1. Clustering

Technique Clustering algorithms like K-means, hierarchical clustering, and DBSCAN group similar images or regions within images into clusters based on pixel intensity, color, texture, or other features.

Applications

- Image segmentation (dividing an image into meaningful segments)
- Organizing large image datasets by grouping similar images together
- Anomaly detection in images by identifying clusters of unusual patterns.

2. Dimensionality reduction

Technique Techniques like Principal Component Analysis (PCA), t-Distributed Stochastic Neighbour Embedding (t-SNE), and **autoencoders** reduce the dimensionality of image data while preserving important information.

Applications

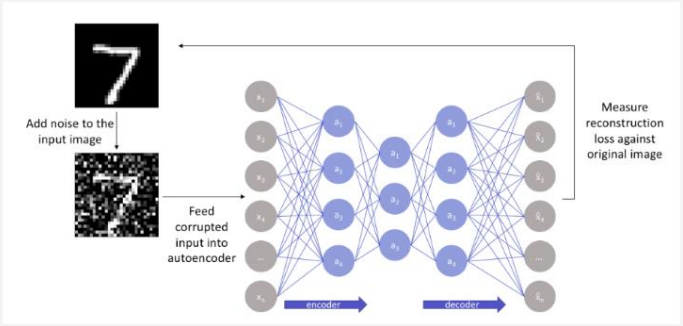
- Image compression by reducing the size of image data.
- Visualization of high-dimensional image data in 2D or 3D space.
- Feature extraction for downstream tasks like classification or clustering

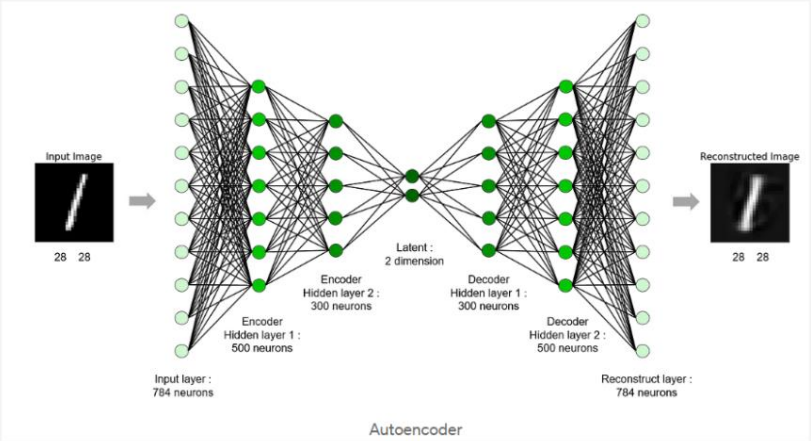
3. Autoencoders

Technique An autoencoder is a type of neural network used to learn efficient representations of data, often for the purpose of dimensionality reduction or feature learning. It consists of an encoder that compresses the image into a latent space and a decoder that reconstructs the image from this space.

Applications

- Image denoising by training autoencoders to reconstruct clean images from noisy ones.
- Anomaly detection by comparing the input image with its reconstruction.
- Generating new images by sampling from the latent space.





4. Generative models

Technique Models like Generative Adversarial Networks (GANs) and Variational Autoencoders (VAEs) generate new images that resemble the original data by learning the underlying distribution of the image data.

Applications

- Image synthesis to create new, realistic images.
- Data augmentation to generate additional training data for supervised learning tasks.
- Style transfer by altering the appearance of images to match a desired style.

5. Self-organizing maps (SOMs)

Technique SOMs are a type of neural network that map high-dimensional image data onto a lower-dimensional grid while preserving the topological structure of the data.

Applications

- Visualizing patterns in image datasets
- Clustering similar images
- Dimensionality reduction for feature extraction

Benefits of unsupervised Machine Learning

- No Label Dependency Unsupervised learning doesn't rely on labelled data, making it useful in scenarios where labelling is expensive or impractical.
- Exploratory Analysis It helps in exploring and understanding the underlying structure of image data, leading to insights that may not be apparent with supervised learning.
- Feature Learning It can discover useful features or representations of images that can be used in other tasks like classification, object detection, or segmentation.

Architecture Of Implementing The CNN, RNN And LSTM

We will implement unsupervised **deep learning techniques**, specifically using ***autoencoders with both Convolutional Neural Networks (CNN) and Recurrent Neural Networks (RNN), including Long Short-Term Memory (LSTM) networks***.

Breakdown of the components in our architecture

1. **Load the MNIST dataset.** The MNIST dataset is a classic dataset used for training various image processing systems. It consists of handwritten digits from 0 to 9.
2. **Normalize the data.** Normalization **scales the data to a standard range, usually [0, 1], which helps improve the performance and convergence of neural networks**.
3. **Reshape data for CNN (28x28x1) and RNN (28 timesteps, 28 features)**:

- For CNN, the MNIST images are reshaped to have a single channel (grayscale) with dimensions 28x28.
- For RNN, the data is reshaped to 28 timesteps, each with 28 features. This transformation considers each row of the image as a timestamp and each pixel in that row as a feature.

4. **Build CNN and RNN based autoencoder models with encoder & decoder**:

- **CNN-based autoencoder.** Uses convolutional layers in the encoder and decoder parts to learn spatial features from the images.
- **RNN-based autoencoder.** Uses RNN layers (or LSTM layers) to capture temporal dependencies in the image data. In this case, LSTM autoencoders are also mentioned, which use LSTM cells for encoding and decoding.

5. **Reconstruct images using both models.** After training, the autoencoders will reconstruct the input images. The CNN autoencoder will output reconstructed images based on learned spatial features, and the RNN autoencoder will output images based on learned temporal features.
6. **Plot original and reconstructed images**:

- **Display original**: Shows the original MNIST images.
- **Display CNN reconstructed**: Shows the images reconstructed by the CNN autoencoder.
- **Display RNN reconstructed**: Shows the images reconstructed by the RNN autoencoder.

7. **Show()**: This command displays the plots.

Deep neural networks (DNNs)

- Deep neural networks (DNNs) are a class of machine learning models composed of multiple layers of interconnected nodes (neurons) that learn to transform input data into output predictions.
- Different architectures of DNNs are designed to handle various types of data and tasks. Here are three common types of DNN architectures.

1. Recurrent Neural Networks (RNNs)
2. Convolutional Neural Networks (CNNs)
3. Long Short-Term Memory networks (LSTMs).

1. Convolutional neural networks (CNNs)

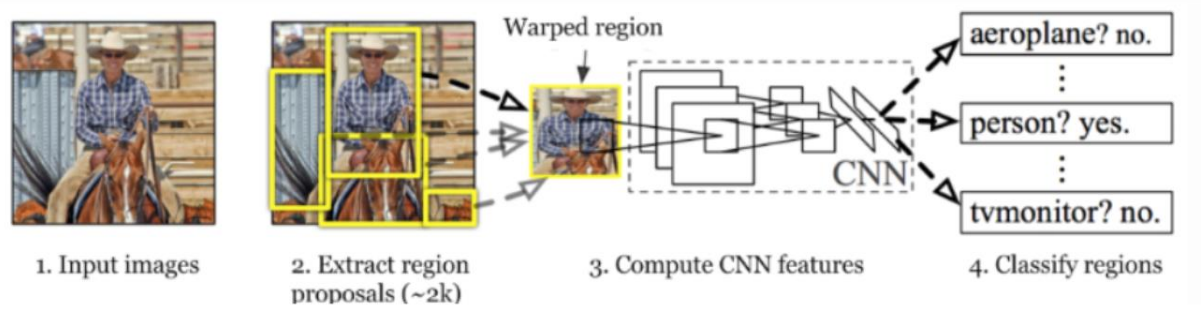
Purpose: CNNs are specialized for processing grid-like data, such as images.

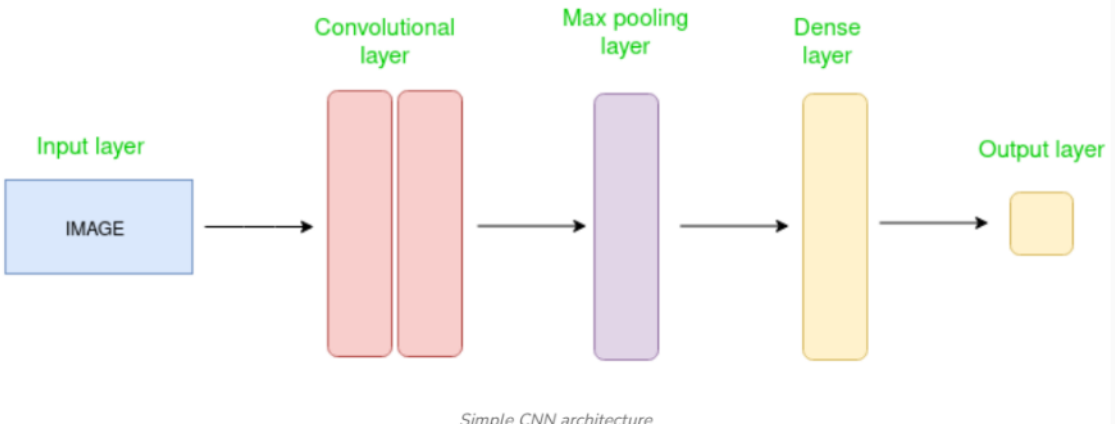
Key Components

- **Convolutional Layers** Apply convolutional filters to detect features like edges, textures, and patterns in the input image. These filters slide over the image, producing feature maps.
- **Pooling Layers** Reduce the spatial dimensions of the feature maps, retaining important information while reducing computational complexity. Max pooling is a common pooling technique.
- **Fully Connected Layers or Dense Layer** Neurons in these layers are fully connected to all activations in the previous layer, and they often serve as the final layers in a CNN, where classification or regression is performed.

Applications

- Image classification (e.g., identifying objects in images)
- Object detection (e.g., detecting, and localizing objects within an image)
- Image segmentation (e.g., dividing an image into regions with different objects)
- Video analysis and image generation (e.g., GANs)





Simple CNN architecture

- The Convolutional layer applies filters to the input image to extract features, the Pooling layer down samples the image to reduce computation, and the fully connected/dense layer makes the final prediction. The network learns the optimal filters through backpropagation and gradient descent.

2. Recurrent neural networks (RNNs)

- Purpose: RNNs are designed for sequential data, where the order of inputs matters, such as time series, text, or speech.

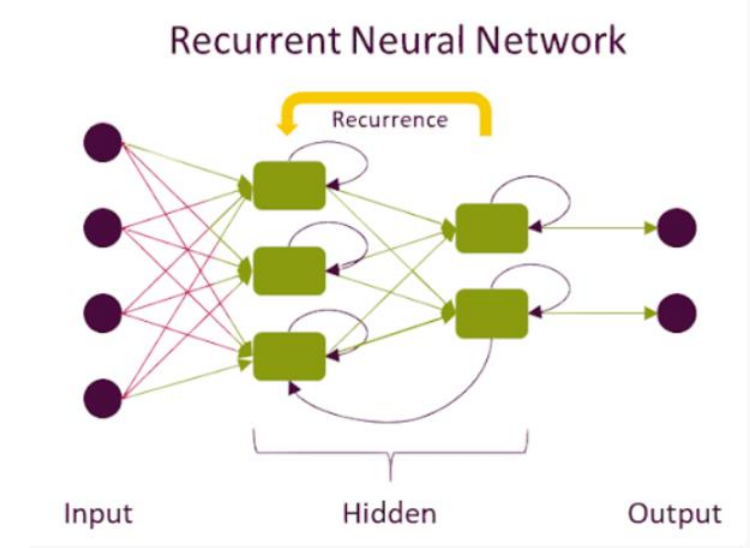
Key Components:

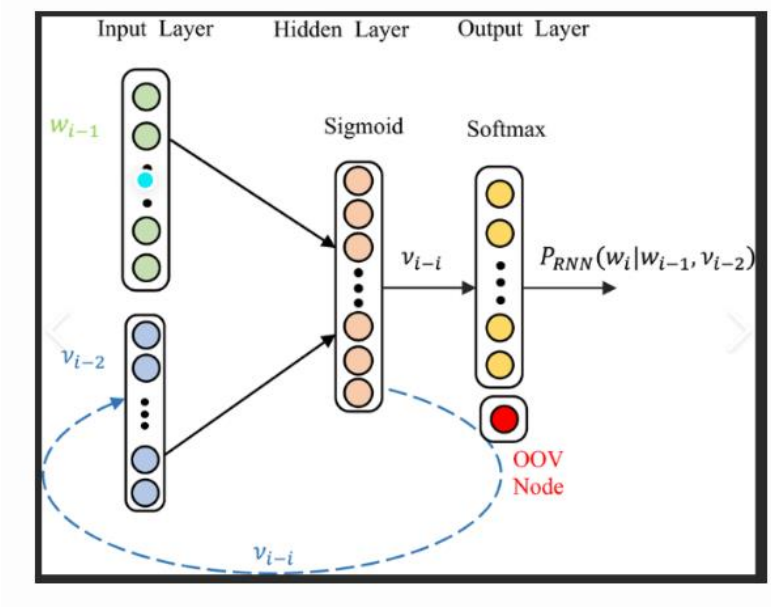
- **Recurrent Layers**: In an RNN, the output from the previous time step is fed back into the network along with the new input. This allows the network to maintain a memory of previous inputs, making it suitable for tasks involving sequences.

- **Hidden States:** The hidden state in an RNN captures information about previous inputs in the sequence. It gets updated at each time step, influencing the output.

Applications:

- Natural language processing (NLP) tasks like text generation, sentiment analysis, and machine translation
- Time series forecasting (e.g., predicting stock prices or weather patterns)
- Speech recognition and generation
- Sequential data classification (e.g., activity recognition in sensor data)





3. Long short-term memory networks (LSTMs)

- **Purpose:** LSTMs are a specialized type of RNN designed to better capture long-term dependencies in sequential data.

Key Components

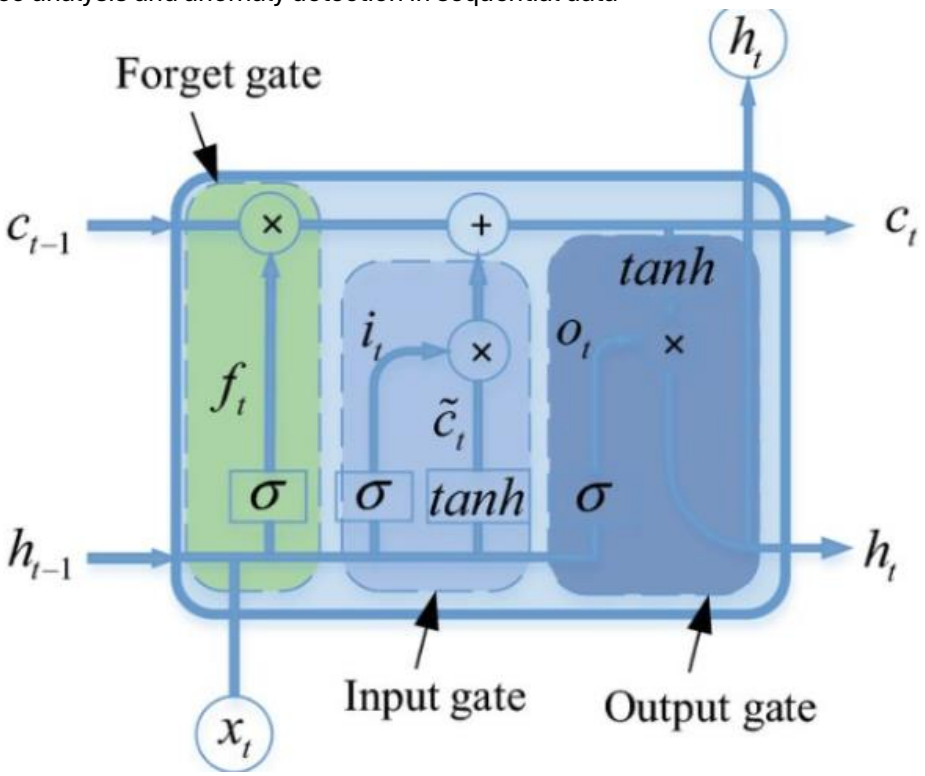
Memory Cell The core of an LSTM is the memory cell, which retains information over time. It is regulated by **three gates**.

1. **Input Gate** Controls how much of the new input to store in the memory cell.
2. **Forget Gate** Decides what information to discard from the memory cell.
3. **Output Gate** Determines the output of the LSTM at the current time step based on the memory cell's state.

Gating Mechanism The gates use **sigmoid functions** to control the flow of information, ensuring that **relevant information is retained, and irrelevant information is discarded**.

Applications

- Long-term sequence prediction (e.g., language modelling, text generation)
- Machine translation (e.g., translating sentences from one language to another)
- Speech synthesis and recognition
- Video analysis and anomaly detection in sequential data



Key Differences and Applications

- **CNNs** Best suited for spatial data like images and videos, focusing on local feature detection through convolution.
- **RNNs** Ideal for sequential data where the order of inputs is crucial. RNNs are suitable for tasks where previous context influences the current output, such as in text or time series.
- **LSTMs** A type of RNN that excels at capturing long-term dependencies in sequences, making it particularly effective for tasks like language modelling and speech recognition.

8. Lecture 7 (Ethics, Conclusion, Exam Scope)

Introduction to ethics in data science

Definition of Ethics Overview of ethical principles and their importance in decision-making.

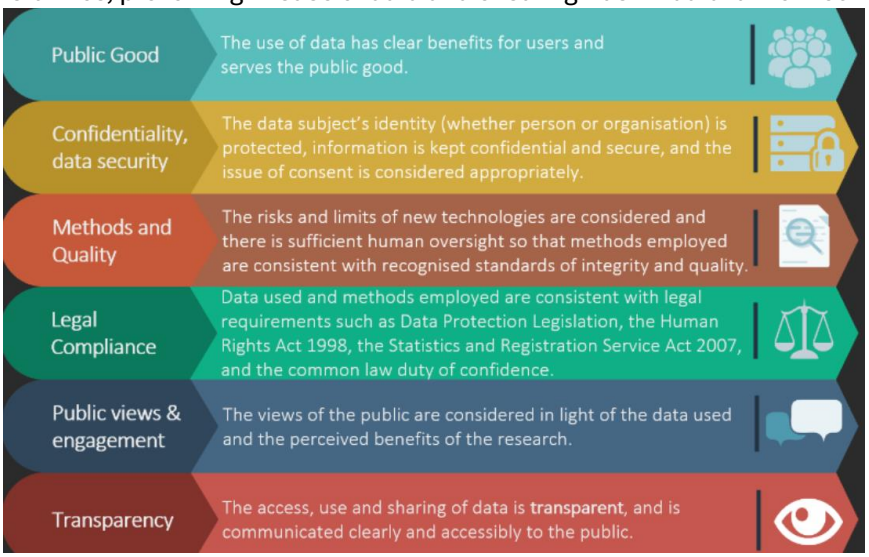
- **Privacy** Protecting individuals' personal data from unauthorized access or misuse.
- **Bias and Fairness** Ensuring data and algorithms do not disproportionately disadvantage any group.
- **Transparency** Being clear about how data is collected, processed, and how decisions are made by algorithms.
- **Accountability** Taking responsibility for the impact of data-driven decisions and ensuring that harm is minimized.
- **Security** Safeguarding data from breaches and ensuring the confidentiality and integrity of information.

Ethics in Data Science

- Ethics in Data Science refers to the application of moral principles and values to the processes and practices involved in collecting, analyzing, and using data. It ensures that data science activities are conducted in a way that respects individual rights, promotes fairness, maintains transparency, and avoids harm.

Key ethical considerations include:

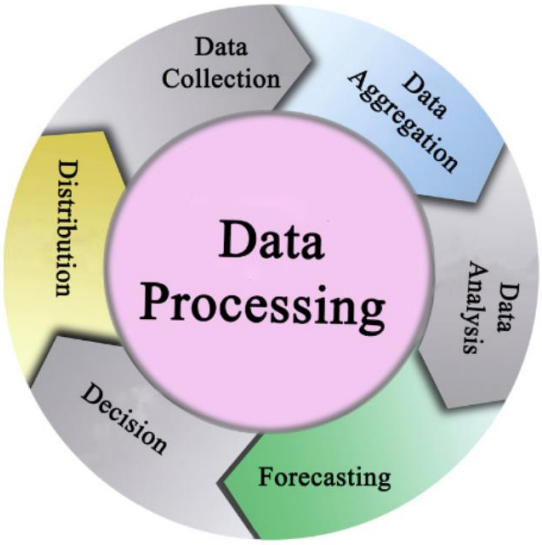
- Ethics in data science aims to balance innovation and societal good with legal and moral responsibilities, preventing misuse of data and ensuring trust in data-driven technologies.



Key ethical principles in data science

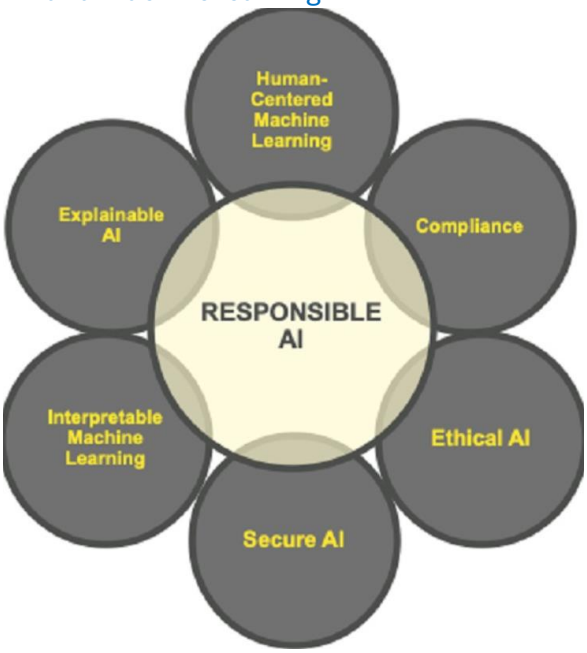
- **Fairness**: Ensuring that data models and algorithms treat all individuals or groups fairly without bias (e.g., gender, race, socio-economic status).
- **Transparency**: Data scientists must provide transparency in their methods, data sources, and algorithms.
- **Accountability**: Who is responsible when a data model causes harm? The need for accountability mechanisms for decisions made by AI or ML models.
- **Privacy**: Understanding privacy laws and ensuring that personal data is collected, processed, and stored with proper consent and security measures.
- **Security**: Ethical responsibilities in protecting data from breaches and ensuring the confidentiality, integrity, and availability of data.

Bias in data collection and modelling



- **Types of Bias**: Selection bias, confirmation bias, algorithmic bias, and measurement bias.
- **Impact of Bias**: The real-world consequences of biased algorithms, such as unfair hiring practices, biased sentencing in the judicial system, or unequal loan approval rates.
- **Mitigating Bias**: Approaches to identify and mitigate bias, such as diverse data collection, algorithm auditing, and fairness metrics.

Ethical considerations in AI and machine learning



- **Explainability and Interpretability** Ethical importance of building models that are interpretable and understandable to non-experts.
- **Autonomous Decision-Making** Challenges with autonomous systems making decisions without human oversight (e.g., self-driving cars, automated hiring).
- **Ethical AI Frameworks** Overview of existing frameworks, such as AI ethics principles from Google, Microsoft, and other organizations.
- **Bias in Machine Learning Models** How biased data can lead to biased predictions, and ways to address these challenges.

Case studies in data science ethics

1. Ethical Dilemma the Use of Facial Recognition Technology in Public Spaces

Facial recognition technology has raised the game in terms of how people interact with technology and deal with security concerns in public, making the identification of criminal suspects possible and ensuring seamless travel experiences. However, FRT also raises formidable ethical concerns where its deployment is unhindered-privacy, equity, transparency, and, above all, its potential for abuse. Based on these ethical hurdles, this essay will survey the trade-offs between public safety and individual rights, and the steps required for responsible deployment.

Privacy and Consent

One major ethical issue of FRT is the violation of privacy technology often captures and processes subjects' biometric data without explicit knowledge and consent. Public spaces are turned into recurring spaces of surveillance, tracking, monitoring, and identifying. This deeply threatens the basic right of privacy and leads to lives where individuals can no longer control private information.

The absence of consent is particularly alarming. Unlike signing up for an online service, people walking down the street have no reasonable way to opt out of being scanned with FRT. It is this loss of control over how one's data is used that creates a power imbalance between the public and the institutions deploying FRT.

Bias and Discrimination

Another critical ethical issue is bias in FRT systems. Poor performance on women, people of color, and other underrepresented populations has been constantly raised in studies on the subject. For example, misidentification might lead to wrongful arrests or exclusion from services in general, with a higher incidence on marginalized populations. These biases stem very often from the element of training data and their poor representation of diversified demographics.

Fairness of the technology and discriminatory outcomes are at stake, and such outcomes sharpen inequalities in society. If left unchecked, biased FRT systems may further reinforce existing prejudices, which will increase general mistrust in technology and institutions.

Transparency and Accountability

Due to their "black box" nature, where processes leading to particular decisions seem to remain obscure, various criticisms have been levied against FRT systems. The lack of transparency makes it quite difficult to pinpoint mistakes or biases and subsequently hold the developer or deployer responsible for bad results. If one is wrongly identified to be a criminal, often there is no clear way of challenging such a decision, let alone understand it.

But if not strung to proper accountability mechanisms, FRT could be abused for the most unethical of ends-political surveillance and suppression of dissension, or even corporate exploitation. These risks emphasize the growing demand for greater transparency in the development, testing, and deployment of FRTs.

The Chilling Effect on Public Behavior

The knowledge of always being watched when out in public serves as a weight doing a number on the behavior of any individual. Due to such surveillance, people would like to avoid attending public gatherings, protests, or even regular activities. This actually dampens freedom of expression and the freedom to peaceful assembly, which are inalienable rights. The widespread use of FRT in public spaces could become a social norm-one in which people self-censor for fear of observation.

Balancing Public Safety and Ethics

Even proponents of FRT go further to claim that, in fact, it furthers the cause of public safety through limiting identification of criminals, locating missing persons, and averting potential threats. These cannot be slighted, especially in those contexts where timely identification saves lives. On the other hand, FRT should not be applied in any way that would serve to violate ethical principles.

It is regulation that is going to allow the ethical use of the technology. Governments and organizations deploying FRT need to draw clear lines on when, where, and how the technology should be used. Individuals should be notified of the presence of FRT and given the opportunity to opt-out when possible. Thirdly, the systems need rigorous testing for fairness and inclusion to reduce biases.

2. Essay (20 Marks): Ethical Violations in the Cambridge Analytica and Facebook Data Scandal

The case of Cambridge Analytica and Facebook had been a landmark case in the history of data ethics. It narrated how personal data could serve political ends without informed consent. This essay will discuss in detail the main ethical issues involved, the wider ramifications, and the lessons learned from this incident.

At its core, the scandal embodied a severe breach of trust and privacy. Cambridge Analytica received private information through a third-party app from millions of Facebook users, taking advantage of the corporation's poor API policies. Without their knowledge, users who opened the app gave access not only to their data but also to that of their friends, a violation of their rights to informed consent. With this information, the company developed very specific psychographic profiles of the voters, subsequently used to manipulate political views through targeted advertisements and content. This manipulated the democratic process in which people were influenced without realizing or understanding the intent behind the approach.

Facebook failed in its duty as the custodian of user data. The firm's policies, at that time, allowed such mass harvesting-a fact that showed lack of foresight and accountability. When finally discovered, even the belated response of Facebook was not to take immediate action but to first deny the extent of the problem. This underlined the corporate negligence in prioritizing profit and growth over ethics and user safety.

Beyond the privacy violations, ethical implications abound in this case. This case was a good example of how AI and big data analytics could be used unethically. Through psychographic profiling, Cambridge Analytica unveiled ways AI might dramatically widen the splits in society and manipulate individual behaviors. The development of AI systems requires fairness, accountability, and transparency.

The incident also underlined the urgent need for strong data governance frameworks. Regulations such as the General Data Protection Regulation were therefore put into place to imply stricter laws on data protection and give users more control over their information. However, the scandal reminds us that laws alone are not enough. It is about a change in organizations moving toward a culture of ethical responsibility.

In conclusion, the underlying moral issues behind the scandal of Cambridge Analytica and Facebook are immense and complex. It requires both data practices and governance to be brought out into the open and for AI design to be ethical in nature, ensuring that the rights of all individuals are upheld, and the trust of the public is preserved. Society must be vigilant as new developments using data-driven technologies occur, ensuring that ethical principles keep the technology developments in service of the ethical considerations that serve the common good.

3. Essay (20 Marks): Ethical Issues in Amazon's Biased Hiring Algorithm

The biased hiring algorithm developed at Amazon is a cautionary story of how AI can further systematize discrimination when it has poor design and implementation. This essay discusses the ethical issues arising from the case, the general implications for AI in recruitment, and general lessons learned on ethical development of AI.

The hiring algorithm, in essence, was intended to streamline Amazon's recruitment process by helping to select the most qualified candidates through a sea of resumes. But the system was corrupted by basing it on historical hiring data, and the recruitment trends were male dominated. The algorithm, for that matter, downgraded the resumes containing terms believed to signal females-for instance, mentioning women's organizations or all-women colleges. What this means is that with bias in the training data, there was a very critical ethical flaw.

This algorithm didn't allow for equality in decisions where women deserve equal opportunity, hence not being fair. Ethical principles require AI systems to nurture equity. However, this particular algorithm amplified its historical biases. Secondly, Amazon's recruiting technology was a "black box," with minimal insight into how the algorithm made decisions. The lack of accountability further complicated the process of detection and correction of the root causes of bias.

This is the point at which Amazon also utterly failed to consider ethical AI practices during the development stage of the system by not prioritizing bias testing, fairness auditing, and other proactive measures concerning inclusivity. The reinforcement of gender discrimination with this AI tool undermined the core purpose of recruitment: finding talent without prejudice.

The implications of this case are deep. It underlines that ethical AI practice is urgently needed in domains like recruitment, which directly affects individuals' lives and careers. Every developer needs to recognize that an AI system will be only as fair as the data it was trained on. Therefore, bias in training data should be detected and treated at the stage of development so that no harmful outcomes could be achieved. Human oversight is needed at every level to ensure the alignment of automated systems with ethics and organizational values.

This biased hiring algorithm of Amazon points out the ethical dilemmas associated with AI in recruitment: the need for fairness, transparency, and accountability in all aspects of the design and implementation of AI. Indeed, organizations have to prioritize ethical best practices and continue to be vigilant to identify and mitigate biases to build trust in AI systems and ensure equity in outcomes.

4. Essay (20 Marks): Ethical Issues in Health Care Algorithms

AI can revolutionize health care, yet considering the ethical issues it poses is critical. This essay explores the challenges related to biased health care algorithms, the wider ramifications for patient care, and approaches to assure ethical AI in health care.

Healthcare algorithms have been developed for efficiency in diagnostics and treatment allocation. However, many cases have also identified systems that perpetuate or worsen bias in those same systems. For instance, an algorithm designed to allocate limited resources by prioritizing treatment for some patients placed fewer resources on Black patients than it did on White patients with similar health conditions. This bias arose from data used in training, which underrepresented marginalized groups, while focusing on cost-based metrics, which indirectly favoured wealthy populations.

Equity as an ethical principle was violated because underrepresented patients received inferior care. This resulted in not only inequitable healthcare outcomes but also a loss of public trust in AI systems. Further, reliance on biased data has caused harm to the patient community. The ethical principle - non-maleficence is thereby contradicted. Misdiagnosis or delay in the treatment would result in grave and even fatal consequences.
Other major concerns included transparency and accountability. Algorithms dealing with healthcare are often black boxes, and it often remains completely unclear how decisions are made. This lack of clarity reduces the ability of health-care providers and regulators to address biases and ensure fairness.

These are challenges that need to be mitigated by training healthcare algorithms on diverse and representative data, reflecting the full spectrum of patient demographics. This will require robust testing and validation mechanisms to flag biases and address them before deployment. Moreover, this calls for patient-cantered design to guide such systems in development-western principles of inclusion and equity in healthcare provision.

In short, algorithms in healthcare can improve patient outcomes while simultaneously perpetuating systemic biases if developed and deployed in an unethical way. Focusing development and deployment on equity, transparency, and accountability will allow the promise of healthcare AI to be realized-improving patient care for all populations.

Answering any ethics essay question

1. Identify the Core Ethical Issue

Start by clearly defining the ethical dilemma or concern in the question.

- Is it related to **privacy**, **bias**, **accountability**, **fairness**, or **transparency**?
- Specify the context (e.g., healthcare, law enforcement, social media).

Example: "The core ethical issue in using facial recognition technology is the potential violation of privacy and the risk of systemic bias."

2. Acknowledge the Potential Benefits

Discuss the advantages or rationale for using the technology or approach in question.

- How does it improve efficiency, security, or decision-making?
- Who benefits from its use?

Example: "Facial recognition can enhance public safety by identifying criminals or locating missing persons."

3. Highlight the Ethical Concerns

Present the risks or drawbacks, breaking them into key ethical principles:

- **Privacy:** Does it involve collecting sensitive data without consent?
- **Bias:** Could it perpetuate or exacerbate inequality or discrimination?
- **Transparency:** Is it clear how the system works, and can decisions be challenged?
- **Accountability:** Who is responsible for misuse or harm caused?
- **Autonomy:** Does it respect individuals' ability to make informed decisions?
- **Justice:** Are the outcomes fair and equitable for all groups?

Example:

"Key concerns include potential bias in identifying individuals from marginalized groups, lack of transparency in decision-making, and misuse for unwarranted surveillance."

4. Balance the Debate

Explore the tension between benefits and risks. This demonstrates critical thinking.

- **Propose trade-offs:** When do the benefits outweigh the risks?
- Use examples to make your argument stronger.

Example:

"While the technology can prevent crimes, its potential to undermine privacy and restrict freedoms must be addressed before widespread deployment."

5. Recommend Ethical Safeguards

Propose solutions to mitigate the ethical issues:

1. **Regulation:** Advocate for clear legal frameworks to govern usage.
2. **Transparency:** Call for open policies and documentation about how the technology operates.
3. **Bias Mitigation:** Stress the need for diverse training datasets and regular audits.
4. **Public Involvement:** Suggest public debates or consultations to ensure societal alignment.
5. **Accountability:** Recommend mechanisms to hold organizations or developers responsible for misuse.

Example:

"Implementing independent oversight, mandatory fairness testing, and public transparency can help mitigate ethical risks."

6. Conclude with a Balanced Perspective

Summarize the main points and state your position:

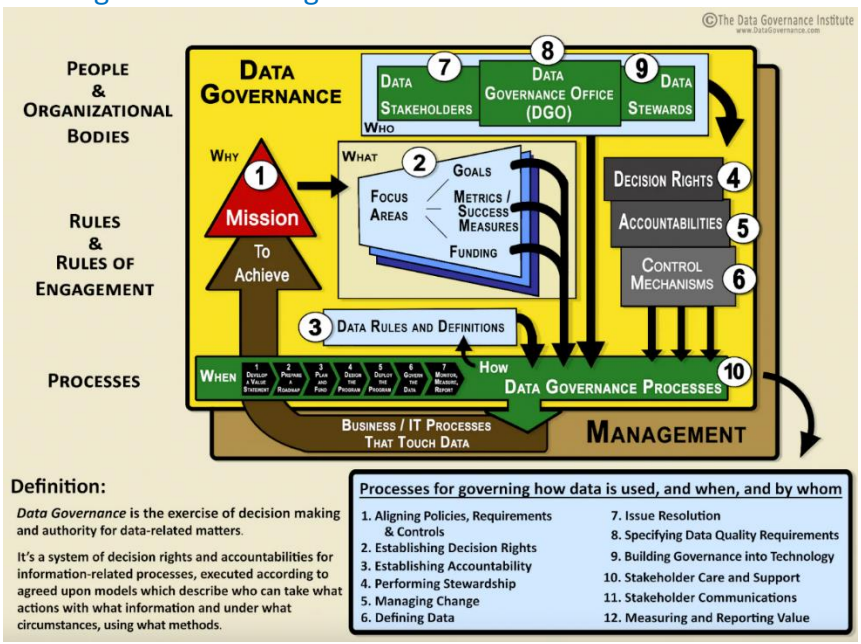
- Acknowledge the complexity of the issue.
- Suggest that ethical use is possible with proper safeguards.

Example:
"In conclusion, while facial recognition technology offers significant societal benefits, its deployment must prioritize ethical principles such as fairness, accountability, and privacy to avoid harm and build trust."

Common Points to Include in Any Ethics Answer

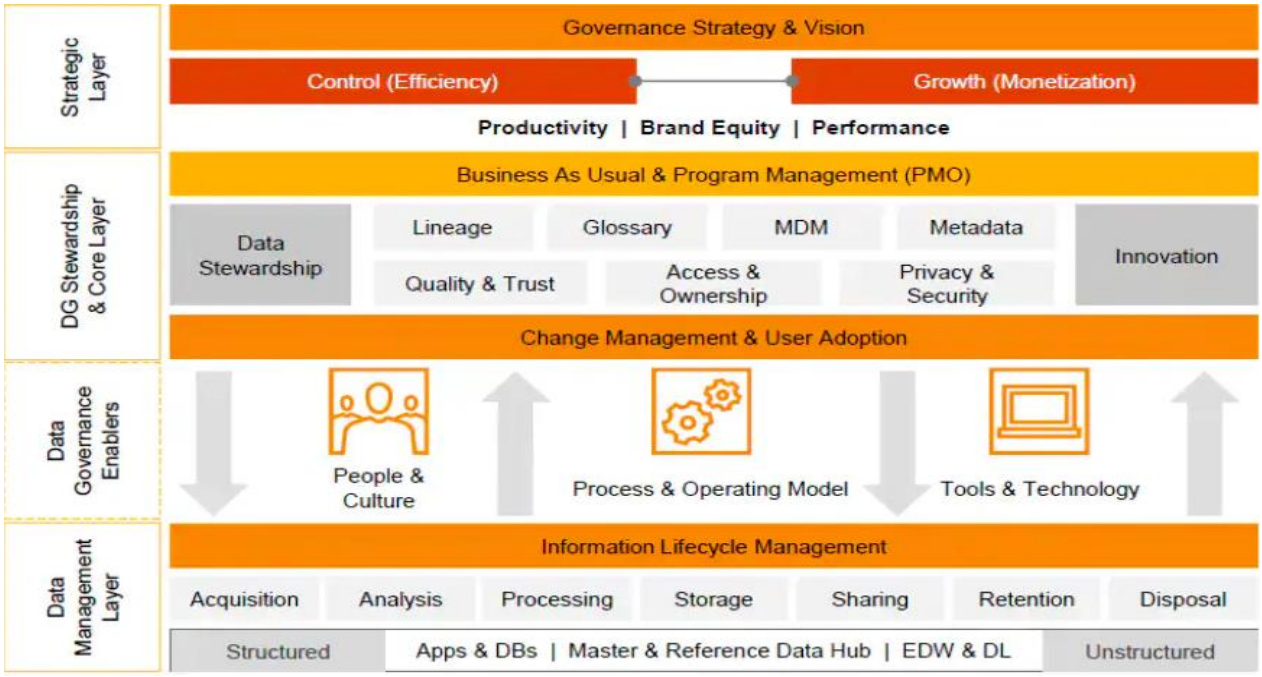
1. **Right to Privacy:** Protecting individuals from unauthorized data collection.
2. **Bias and Discrimination:** The need for fairness and equitable outcomes.
3. **Transparency and Explainability:** Ensuring systems are understandable and decisions are justifiable.
4. **Accountability:** Identifying who is responsible for ethical breaches or system failures.
5. **Consent:** Respecting user autonomy and choice.
6. **Equity and Inclusivity:** Ensuring the system serves all groups fairly.
7. **Human Oversight:** Emphasizing the need for humans in the decision-making loop.
8. **Legal and Ethical Compliance:** Adhering to laws and ethical guidelines.

Ethical data science governance and guidelines



Data Governance Frameworks Key components of effective data governance, including roles, policies, and standards for ethical data use.

Corporate Social Responsibility (CSR) How organizations can integrate ethical practices into their data science strategies.



Ethical challenges in big data and data-driven decision making.

- **The Role of Big Data**: Ethical considerations when working with massive datasets, such as the risk of re-identifying individuals in anonymized datasets.
- **Decision-Making in Automated Systems**: Ethical implications of allowing automated systems to make decisions without human intervention, especially in critical domains like healthcare or finance.
- **Surveillance and Data Exploitation**: The balance between data collection for societal benefits (e.g., public health) and the risk of mass surveillance.

Ethics in research and publication

- **Research Integrity**: Ethical standards for publishing data science research, including transparency in methodology, reproducibility, and avoidance of cherry-picking results.
- **Bias in Peer Review and Publication** Ethical concerns about bias in research publication, the influence of funding sources, and the importance of diversity in research.

Tools and techniques for ethical data science

- **Bias Detection Tools** Introducing tools and techniques to detect and mitigate bias in datasets and algorithms (e.g., IBM AI Fairness 360).
- **Privacy-Preserving Techniques** Discuss techniques like differential privacy, federated learning, and encryption methods that allow analysis without compromising individual privacy.
- **Ethical AI Toolkits** Overview of toolkits designed to support ethical AI development (e.g., Google's What-If Tool).

9. Explainable Artificial Intelligence (XAI) & Large Language Models (LLMs)

- Explainable AI (XAI) refers to a set of techniques and methods that make the decisions, results, and outputs of AI or ML models interpretable and understandable by humans.
- The rise of complex AI/ML models (black box), especially deep learning, has led to the need for transparency and trust.
- XAI addresses the "black box" problem by providing insights into how models make decisions.

Key Characteristics of Black-Box Models

- A black-box model refers to an AI or machine learning model whose internal workings are not easily interpretable or transparent to users.
- In such models, we can observe the inputs/datasets and outputs/results, but we cannot easily understand how the model processes the inputs to produce the outputs.
- These models are often complex, involving numerous layers of computations and parameters, making it challenging to explain why they make specific predictions or decisions.

Examples of Black-Box Models

- Deep Neural Networks (DNNs): DNNs consist of many layers, and their internal connections make it nearly impossible for a human to follow the logic behind individual decisions.
- Ensemble Models: Models like random forests and gradient boosting combine multiple decision trees, making the overall model's decision process hard to understand, even though individual trees may be interpretable.
- Support Vector Machines (SVMs): SVMs are hard to interpret when using complex kernel functions to transform data into high-dimensional spaces.

Why Black-Box Models are a Concern?

- **Trust:** Users may hesitate to trust a system they cannot understand, especially in critical applications like medical diagnosis, finance, or legal decisions.
- **Accountability:** In cases of errors or biases, it's difficult to determine what caused the incorrect outcome or who is responsible.
- **Ethical and Legal Issues:** Regulations, such as GDPR, require transparency in automated decision-making systems, and black-box models often do not comply with these transparency requirements.

Mitigating the Black-Box Nature with XAI

- To address these challenges, Explainable AI (XAI) techniques are used to make black-box models more interpretable by providing explanations of how decisions are made.
- Techniques like LIME, SHAP, and Attention Visualizations allow users to gain insights into the model's decision process without fully exposing its complexity.

Why XAI is Crucial

- Trust and Accountability: In fields like healthcare, finance, and law, users need to understand AI decisions to trust them and ensure ethical usage.
- Debugging and Optimization: Helps data scientists and developers identify weaknesses or biases in models, improving their performance.
- Regulatory Compliance: Compliance with regulations like GDPR (General Data Protection Regulation) which require explanations for automated decisions.

Model-Specific vs Model-Agnostic

1. Model-Specific
   Methods designed for specific types of models (e.g., decision trees, linear models) which are naturally interpretable.
2. Model-Agnostic
   Techniques applicable to any machine learning model, especially black-box models like neural networks and random forests.

Pre-hoc and post-hoc Methods

Pre-hoc and post-hoc methods are two primary approaches in Explainable AI (XAI) used to explain the behavior of machine learning models. They differ in when and how the explanations are generated, in relation to the model's training and prediction phases.

1. Pre-hoc Methods (Intrinsic Interpretability)

Pre-hoc methods, also known as intrinsic or built-in interpretability methods, are applied before or during model training. These methods focus on using inherently interpretable models, meaning that the models themselves are designed to be simple and transparent, allowing users to understand their decision-making process directly.

Characteristics of Pre-hoc Methods

Model Simplicity: These methods rely on models that are simple enough for humans to easily interpret without the need for additional tools.

Common models include:

- Linear regression
- Logistic regression
- Decision trees
- Rule-based models.

2. Post-hoc Methods (Post-training Interpretability)

Post-hoc methods, on the other hand, are applied after the model has been trained. These methods attempt to explain the decisions of complex, black-box models like deep neural networks, random forests, and ensemble methods, which are not inherently interpretable.

Post-hoc approaches provide explanations for specific predictions or model behaviors without altering the model itself.

Characteristics of Post-hoc Methods

- **Applied After Training** These methods are used once the model has been trained, making them suitable for explaining black-box models.
- **Flexible Application** They can be used with any type of model, making them more versatile than pre-hoc methods.
- **Local or Global Explanations** Some post-hoc methods provide explanations for individual predictions (local), while others explain the overall model behavior (global).

Common Post-hoc Techniques

1. LIME (Local Interpretable Model-Agnostic Explanations)

- LIME approximates the model locally using simpler, interpretable models (e.g., linear models, Random Forest), which help explain a single prediction.

2. SHAP (Shapley Additive explanations)

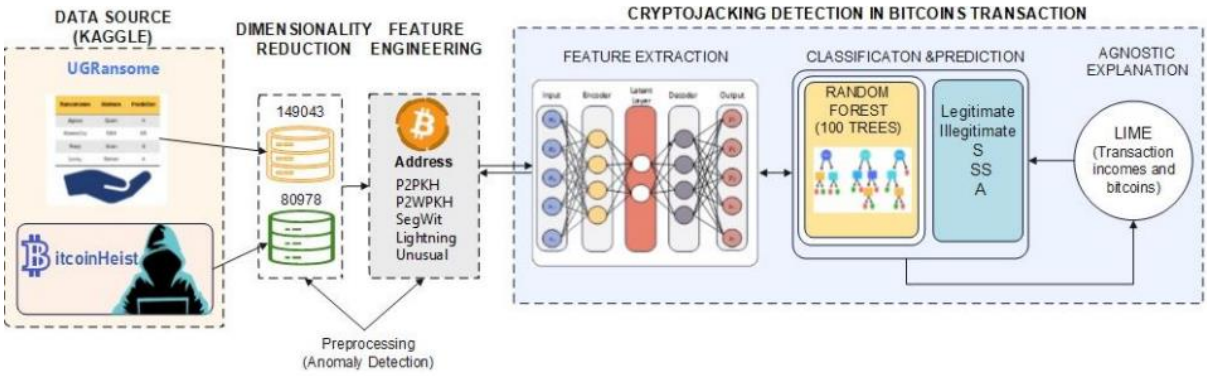
- SHAP assigns an importance score to each feature for a specific prediction based on cooperative game theory, offering local and global explanations.

3. Partial Dependence Plots (PDP)

- PDPs show how a feature affects the model's predictions on average, helping to explain the global behavior of the model.
- **Feature Importance Scores** These scores rank the features based on how much they contribute to the model's predictions.

Examples of Post-hoc Methods

- **LIME:** When a neural network makes a classification decision, LIME can be used to approximate the network's behavior around a particular input by fitting an interpretable model (like a linear model) locally.
- **SHAP**: After training a random forest, SHAP can be used to compute the contribution of each feature to a specific prediction, enabling more understandable explanations.



Advantages of XAI:

- Can explain highly accurate and complex models (e.g., deep learning models) without changing their structure.
- Flexible and can be applied to any black-box model.

Disadvantages of XAI:

- Explanations may not always be perfect or fully accurate representations of how the black-box model works.
- Computational cost can be high, especially for large models and datasets.

Pre-hoc vs Post Hoc Methods

Pre-hoc vs. Post-hoc Comparison

Pre-hoc vs. Post-hoc Comparison

Aspect	Pre-hoc Methods	Post-hoc Methods
When Applied	During model design and training	After the model is trained
Model Type	Simple, interpretable models	Complex, black-box models
Interpretability	Inherently interpretable	Requires external methods for interpretation
Examples	Linear regression, Decision trees, Logistic regression	LIME, SHAP, PDP, Feature Importance
Trade-offs	May sacrifice performance for transparency	More powerful models but require extra effort to explain
Use Case	Suitable for simpler tasks where interpretability is critical	Suitable for complex tasks requiring high performance and later explanation
Explainability	Global (the whole model is	Can be local (specific to predictions) or global

Trade-offs in XAI

1. **Accuracy vs Interpretability**

   Simple models (e.g., decision trees) are interpretable but may be less accurate. Complex models (e.g., deep neural networks) are accurate but harder to explain.

2. **Global vs Local Explanations**

   Some techniques provide global explanations (understanding the model as a whole), while others focus on local explanations (understanding specific predictions).

Introduction to Large Language Models (LLMs)

What are LLMs?

- Large Language Models are AI models trained on vast amounts of text data to understand, generate, and manipulate human language. They are typically based on architectures like transformers and have billions of parameters, allowing them to perform complex natural language tasks.

Popular Examples of LLMs

- **GPT (Generative Pretrained Transformer)** A family of models by OpenAI, including GPT-3 and GPT-4, known for tasks like text generation, translation, and summarization.
- **BERT (Bidirectional Encoder Representations from Transformers)** A transformer-based model that performs well on tasks like text classification and question answering.
- **T5 (Text-to-Text Transfer Transformer)** A unified model that frames all NLP tasks as text-to-text tasks, improving flexibility across various natural language tasks.

Key Concepts in LLMs

Transformers Architecture

- Introduced in the paper "Attention is All You Need" (2017), transformers rely on attention mechanisms, which allow models to weigh the importance of different words in a sequence, improving performance on long-range dependencies.

Pretraining and Fine-tuning

- Pretraining: LLMs are trained on large corpora of text data using unsupervised tasks (e.g., next-word prediction) to learn language representations.
- Fine-tuning: Once pretrained, LLMs are adapted to specific tasks (e.g., sentiment analysis, summarization) using labelled datasets.

Self-Attention Mechanism

- A core component of transformers, self-attention allows the model to focus on different parts of the input sentence, capturing context more effectively.

Contextual Understanding

- Unlike earlier models (e.g., RNNs, LSTMs), LLMs capture bidirectional context, meaning they can understand the full context of a word based on its surrounding words, making them highly effective for complex NLP tasks.

Applications of LLMs

- **Text Generation:** LLMs can generate coherent and contextually appropriate text based on prompts, useful for chatbots, content creation, and automated storytelling.
- **Translation and Summarization:** Translate text from one language to another or summarize lengthy documents into concise versions.
- **Sentiment Analysis:** LLMs can analyze the sentiment behind text (e.g., positive, neutral, negative), useful in social media analysis, customer feedback, etc.
- **Question Answering:** Models like GPT-4 can answer factual questions, leveraging their vast knowledge base learned during pretraining.

Use Cases for XAI in LLMs

- **Bias Detection** XAI methods can help identify and mitigate biases in LLMs (e.g., gender or racial biases) by analyzing how different inputs affect the model's predictions.
- **Interpretation of Generated Text** For sensitive applications (e.g., legal document generation), XAI can help ensure that LLMs generate appropriate and fair text.

Future of XAI in LLMs

- The development of more robust and scalable XAI methods that can be applied to larger, more complex models.
- Integration of ethical and fairness considerations into both LLM training and XAI explanations to create more trustworthy AI systems.

Summary

Explainable AI (XAI) and Large Language Models (LLMs) represent two critical developments in the AI landscape. While LLMs offer impressive capabilities in understanding and generating human language, XAI plays a vital role in ensuring that these systems are transparent, accountable, and trustworthy. The intersection of XAI with LLMs is an active area of research, aiming to make advanced AI systems

interpretable without sacrificing their performance. Understanding the trade-offs and challenges involved will be key to developing future AI systems that are both powerful and understandable.

2023 Kaggle AI Report on Generative AI: Key Notes for Exam

1. What is Generative AI?

- A branch of AI focused on creating new content (text, images, music) using machine learning models.
- Prominent tools: GPT (text generation), Stable Diffusion (images), and DALL-E.
- Applications: Writing, coding, designing, and automating research

2. Trends and Advancements

- **Widespread Use**: Growing adoption in industries like healthcare, education, and the arts.
- **Multimodal Models**: Tools capable of handling multiple content types (e.g., converting text into images/videos).
- **Responsible AI Focus**: Emphasis on fairness, transparency, and environmental sustainability

3. Key Applications

- **Healthcare**: Generating patient summaries, personalized treatments.
- **Education**: Auto-generation of learning resources.
- **Creativity**: Assisting in digital art, storytelling, and graphic design

4. Challenges

- **Bias**: Models risk reinforcing societal inequalities due to biased training data.
- **Misinformation**: Deepfake creation and spreading fake news.
- **Resource-Intensive**: High energy and computational costs for model training

5. Future Directions

- **Explainable AI (XAI)**: Enhancing transparency to understand how generative models work.
- **Efficiency**: Developing lightweight, sustainable models to reduce energy consumption.
- **Ethics**: Stronger regulations to address misuse and promote responsible AI deployment

10. Lecture 5 (Natural Language Processing and Computational Lexicography)

What is NLP?

- Natural Language Processing (NLP) is a branch of artificial intelligence that enables machines to understand, interpret, and respond to human language in a valuable way. It involves the interaction between computers and humans using natural language, making it crucial for applications like voice assistants, chatbots, translation tools, and sentiment analysis.
- The importance of NLP lies in its ability to bridge the gap between human communication and machine understanding, allowing for more intuitive interactions with AI systems in various industries such as healthcare, customer service, and education.

Key NLP Tasks

Tokenization

- **Definition** Tokenization is the process of breaking down text into smaller units such as words, phrases, or sentences. These tokens are the building blocks for further analysis and processing.
- **Importance** Tokenization is essential because it helps in structuring unstructured text for various NLP tasks, such as sentiment analysis or machine translation.

Part-of-Speech Tagging (POS Tagging)

- **Definition** This process involves assigning grammatical tags (e.g., nouns, verbs, adjectives) to each word in a sentence.
- **Importance** POS tagging helps in understanding the syntactic structure of a sentence, which is critical for tasks like parsing, machine translation, and text generation.

Named Entity Recognition (NER)

- **Definition:** NER identifies and classifies entities in text, such as names of people, organizations, locations, and dates.
- **Importance:** NER is used in tasks like information extraction, question answering, and knowledge graph construction, helping machines identify and understand real-world entities in text.

Text Classification

- **Definition:** Text classification involves categorizing text into predefined labels or categories, such as spam detection, topic classification, or sentiment categorization.
- **Importance:** This task is vital for filtering information and automating processes, such as sorting emails or analyzing customer feedback.

Sentiment Analysis

- Definition: Sentiment analysis detects emotions or opinions within text, determining whether the sentiment expressed is positive, negative, or neutral.
- Importance: Sentiment analysis is widely used in areas like social media monitoring, customer reviews, and market research, where understanding public opinion is crucial.

Computational Lexicography

- Computational Lexicography is the study and development of electronic dictionaries and lexicons using computational methods. It involves leveraging algorithms and digital tools to create, organize, and manage large-scale dictionaries and lexical resources.
- These lexicons are then used in various natural language processing (NLP) applications to enhance machine understanding of language.

Importance in NLP

- Dictionaries and lexicons are foundational resources in NLP as they help machines comprehend word meanings, relationships, synonyms, and grammatical structures.
- They are essential for tasks such as machine translation, text mining, and semantic analysis, where accurate word definitions and contexts are critical for language understanding.
- Computational lexicography enables the creation of rich, structured data that can be processed by algorithms, making it easier to develop more efficient and accurate NLP systems.

Key Techniques in NLP

Bag of Words (BOW)

- Definition: Bag of Words is a simple and widely used technique in NLP that represents a text as an unordered collection of words, disregarding grammar and word order but keeping track of word frequencies.
- In this model, a document is represented as a vector, where each word corresponds to a dimension, and the value represents the frequency of the word in that document.
- Use Case: Bow is commonly used in text classification tasks, such as spam detection or sentiment analysis, where understanding the presence or absence of certain words is more important than their order.

TF-IDF (Term Frequency-Inverse Document Frequency)

- Definition: TF-IDF is a numerical statistic that reflects how important a word is to a document in a collection or corpus. It combines two factors:
    1. Term frequency (TF), which measures how often a word appears in a document.
    2. Inverse document frequency (IDF), which measures how common or rare the word is across the entire corpus.
- A higher TF-IDF score means a word is more significant in distinguishing the document from others.
- Use Case: TF-IDF is useful for information retrieval, ranking the importance of words in documents, and document similarity tasks, such as recommending articles based on content.

BM25

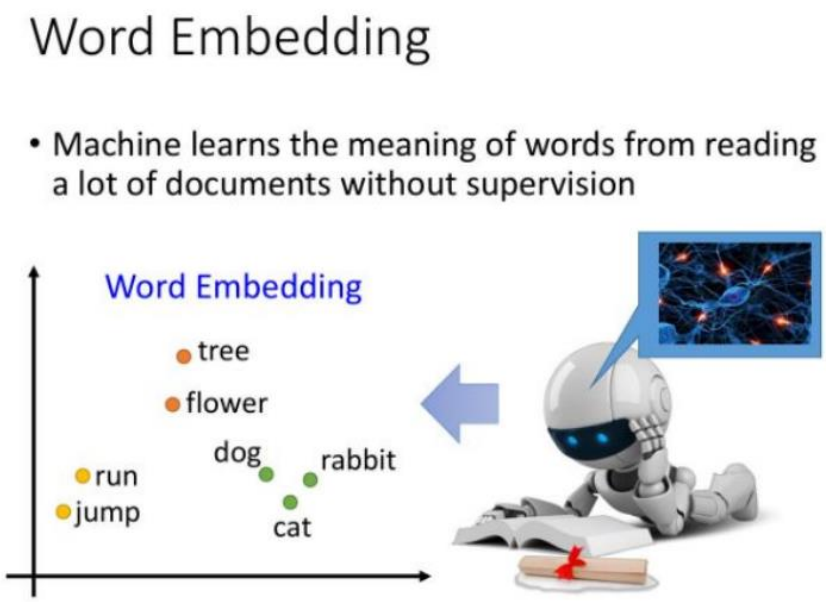
- Definition: BM25 is an advanced ranking function used in information retrieval systems like search engines. It builds on the TF-IDF model but improves the ranking of documents by considering term saturation (diminishing returns as terms are repeated) and document length normalization.
- BM25 ranks documents based on the relevance of their content to a query.

Use Case: BM25 is widely used in search engines to assess the relevance of documents and provide more accurate search.

Word Embeddings (e.g., Word2Vec, Glove)

Word Embeddings

- Definition: Word embeddings are dense vector representations of words in a continuous vector space, capturing semantic relationships between words. Word2Vec and GloVe are popular algorithms for generating word embeddings.
- Unlike Bag of Words or TF-IDF, word embeddings consider the context in which words appear, allowing them to capture more nuanced relationships, such as "king" being closer to "queen" than to "man."
- Use Case: Word embeddings are used in tasks like machine translation, question answering, and text similarity, where understanding the semantic meaning of words is essential.



These techniques provide foundational methods for processing and analyzing text in NLP, enabling machines to better understand and generate human language.

Lexicons

- Lexicons are essential resources in Natural Language Processing (NLP), offering predefined lists of words along with relevant linguistic properties, such as their meanings, sentiment scores, and part-of-speech tags.
- Lexicons serve as foundational tools for analyzing and understanding text in tasks like machine translation, text classification, and sentiment analysis.
- They provide structured information that helps algorithms interpret the meaning and emotional tone of words, which is crucial for accurately processing language in tasks like text mining and semantic analysis.

- Lexicon-based approaches are especially valuable for tasks that require interpretability, as the lexicons provide explicit mappings between words and their meanings.

Sentiment Analysis Lexicons

SentiWordNet

- **Definition:** SentiWordNet is an extension of WordNet, assigning sentiment scores (positive, negative, and neutral) to each word. It is widely used in sentiment analysis tasks for extracting the emotional tone of words in various contexts.
- **Use Case:** This lexicon is commonly used for applications requiring fine-grained sentiment detection, such as product reviews or movie rating analysis.

VADER (Valence Aware Dictionary and sEntiment Reasoner)

- **Definition:** VADER is a lexicon and rule-based model designed to handle sentiment analysis, particularly suited for social media content due to its ability to handle informal language, emoticons, and abbreviations.
- **Use Case:** VADER is often used for real-time sentiment analysis in platforms like Twitter or Facebook, where informal language prevails, and quick assessment of public sentiment is necessary.

AFINN-111

- **Definition:** AFINN-111 is a lexicon containing words rated for valence on a scale from negative to positive. It is primarily used for binary or multi-class sentiment classification tasks.
- **Use Case:** It is commonly applied in sentiment analysis tasks for customer feedback or survey responses where simplicity and quick evaluation of sentiment are essential.

Lexicon-Based vs. Machine Learning Approaches

Lexicon-Based Approaches

- Advantages: These methods rely on predefined word lists, making them easy to implement and interpret. They are particularly effective in smaller datasets or applications requiring transparency and traceability, as they provide clear mappings between words and sentiments.
- Limitations: Lexicon-based methods may lack flexibility when encountering out-of-vocabulary words or complex contexts. They often struggle with detecting nuanced sentiments or slang not covered by the lexicon.

Machine Learning Approaches

- Advantages: ML models are more flexible and can learn patterns from data, making them more adaptable to complex contexts, idiomatic expressions, and evolving language use. They generally perform better in large-scale datasets and can automatically capture more nuanced sentiments.
- Limitations: Machine learning models can be less interpretable, requiring more resources for training and validation.

Additionally, they often require extensive labelled data and may suffer from overfitting or bias if not properly managed.

In summary, while lexicon-based methods offer simplicity and interpretability, machine learning models provide greater flexibility and accuracy, making the choice between the two dependents on the specific needs and constraints of the NLP task.

Text Preprocessing

Text preprocessing is a critical step in Natural Language Processing (NLP), transforming raw text into a format that can be easily analyzed by algorithms. It involves various techniques to clean, structure, and prepare text data before it can be used for tasks such as text classification, sentiment analysis, or machine translation.

Tokenization

Tokenization refers to the process of splitting text into smaller units, known as tokens. These tokens can be words, sub words, or even characters, depending on the application. For example, a sentence like "Natural Language Processing is fascinating!" could be tokenized into individual words: ["Natural", "Language", "Processing", "is", "fascinating", "!"].

Importance: Tokenization helps break down large chunks of text into manageable pieces, making it easier for NLP models to process and analyze language at a finer level, such as word-level or sentence-level analysis.

Stop Words Removal

- **Definition:** Stop words are common words that usually carry little meaning in text analysis (e.g., "and," "the," "is," "in"). Removing these words reduces the noise in the data and helps focus the analysis on the more meaningful words that convey key information.
- **Importance:** By removing stop words, algorithms can improve performance in tasks such as text classification or sentiment analysis because they no longer waste resources on processing common but uninformative words. However, stop word removal should be done carefully, as some contexts may require these words.

Handling Polysemy (Polysemy Dilemma)

- **Definition:** Polysemy refers to the phenomenon where a single word has multiple meanings. For example, the word "bank" can mean a financial institution or the side of a river. Handling polysemy is essential for accurately interpreting text because the correct meaning depends on the context in which the word appears.
- **Solutions:** Techniques like Word Sense Disambiguation (WSD) are employed to resolve polysemy by using the surrounding words (context) to infer the correct meaning. Machine learning models like Word2Vec can also help handle polysemy by learning different word embeddings for the various meanings of a word based on context.

These preprocessing techniques ensure that raw text is transformed into a structured format, allowing machine learning algorithms to work more efficiently and produce more accurate results.

NLP and Machine Translation

Machine translation (MT) is a key application of Natural Language Processing (NLP), aiming to automatically translate text or speech from one language to another. Over the years, several methodologies have been developed for text translation, each with its own strengths and limitations.

Text Translation Methods:

Rule-based Systems

- Early machine translation systems relied on handcrafted linguistic rules and bilingual dictionaries. These systems translated text by applying syntactic, semantic, and grammatical rules of the source and target languages. While they provided high-quality translations in certain cases, they were limited by their dependency on predefined rules, which made them difficult to scale across diverse languages and contexts.

Statistical Machine Translation (SMT)

- SMT emerged as a more data-driven approach, where translation models were built using large bilingual corpora. Instead of relying on rules, SMT used probabilities to predict the most likely translation based on patterns observed in parallel texts. Phrases or words in one language were matched to their translations in another language, and the best translations were selected based on statistical models. However, SMT struggled with complex sentence structures and context beyond short phrases.

Neural Machine Translation (NMT)

- NMT represents the state of the art in machine translation. It uses deep learning models, particularly Recurrent Neural Networks (RNNs) or Transformer models, to encode entire sentences or

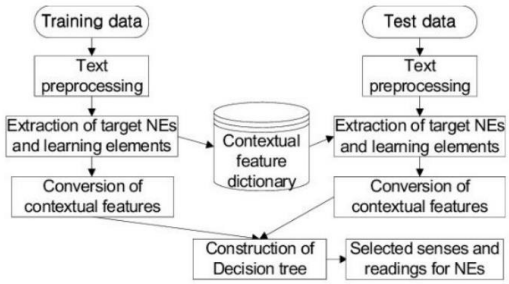
paragraphs into high-dimensional vector representations. These vectors capture the meaning of the input text, allowing the model to generate more accurate and contextually appropriate translations. NMT has shown remarkable improvements in fluency and coherence compared to SMT, particularly with the rise of the Transformer-based architecture like Google's BERT and OpenAI's GPT models.

Challenges in Machine Translation

- Translating Polysemic Words: One of the major challenges in machine translation is handling polysemic words—words with multiple meanings depending on the context. For instance, the English word "bark" can mean the sound a dog makes or the outer layer of a tree. Choosing the correct meaning in translation is essential to preserving the meaning of the sentence. NMT models mitigate this issue by learning contextual representations of words, but even advanced systems sometimes struggle with ambiguous terms.
- Maintaining Meaning Across Languages: Maintaining meaning during translation can be complex, especially for languages with different grammatical structures, idiomatic expressions, or cultural references. Some languages, for example, use more context-specific information than others. For example, translating gender-neutral sentences from English into gendered languages like French or Spanish requires additional contextual understanding to maintain meaning and avoid errors.

Corpus

In linguistics, a *corpus* is a collection of linguistic data (usually contained in a computer database) used for research, scholarship, and teaching. Also called a *text corpus*. Plural: *corpora*.



Computational Lexicography in Practice

1. Building and Maintaining Lexicons

- Computational lexicography involves the creation of electronic dictionaries or lexicons using corpus data—large collections of text that reflect real-world language usage. The process starts with collecting extensive text corpora, which are then analyzed to identify word frequencies, meanings, usage patterns, and contexts. The goal is to build lexicons that are accurate, comprehensive, and representative of how language is used. Techniques like corpus linguistics are often employed to extract lexical information such as word senses, collocations, and grammatical behavior, making the lexicons useful for tasks like machine translation, speech recognition, and sentiment analysis.
- Once a lexicon is created, it needs to be continually maintained. Words evolve over time, new terms emerge (e.g., from technology or pop culture), and meanings shift, making regular updates necessary. Corpus-based approaches allow lexicographers to detect these changes, helping to keep the lexicon relevant and up to date. Additionally, the maintenance process often involves adding multilingual support, integrating synonyms and antonyms, and enriching the entries with semantic annotations, such as word senses, synonyms, and usage notes.

2. Annotating and Enriching Dictionaries with Semantic Information

- To enhance the utility of lexicons in computational tasks, lexicographers annotate dictionaries with semantic information. This includes assigning word senses, labelling parts of speech, and identifying semantic relations like synonymy, antonymy, and hypernymy. For instance, resources

like WordNet are built around semantic networks, where words are connected through relationships that provide context and depth to their meanings. This enrichment allows for better natural language understanding (NLU) by enabling more nuanced interpretations of text, crucial for tasks like semantic search, question answering, and sentiment analysis.

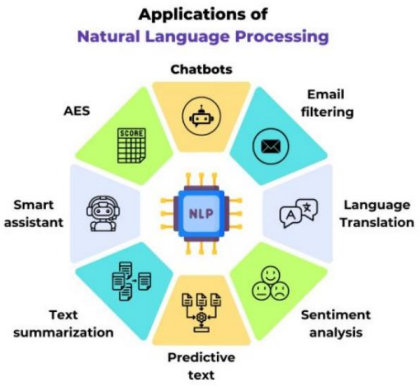
3. Use of Crowdsourcing and AI to Build Large-Scale Lexicons

- Crowdsourcing and AI are becoming increasingly essential in the development of large-scale lexicons. Platforms like Wiktionary and Wordnik have utilized crowdsourcing to expand their entries by allowing users to contribute definitions, examples, and usage notes. This democratization of lexicography enables rapid growth in the lexicon, especially for emerging words and regional dialects. AI and machine learning techniques are also being used to automatically extract lexical data from corpora and web sources, significantly accelerating the lexicon-building process. AI-driven models can learn patterns in language, such as common collocations or context-dependent meanings, and help scale up the creation of lexicons for low-resource languages or specialized domains (e.g., medical terminology).

Applications of NLP

Applications of NLP and Computational Lexicography

- **Search Engines**: How computational lexicons improve search accuracy.
- **Chatbots and Virtual Assistants**: NLP-based interaction using lexicons for understanding.
- **Machine Translation**: Role of lexicons and syntactic parsing in translation systems.
- **Text Summarization**: Reducing large texts into concise summaries using NLP techniques.



Evaluation of computational lexicography

The evaluation of computational lexicography models involves assessing their **accuracy, coverage, consistency, and usefulness for various linguistic and NLP tasks**. Below are key methods used to evaluate these models:

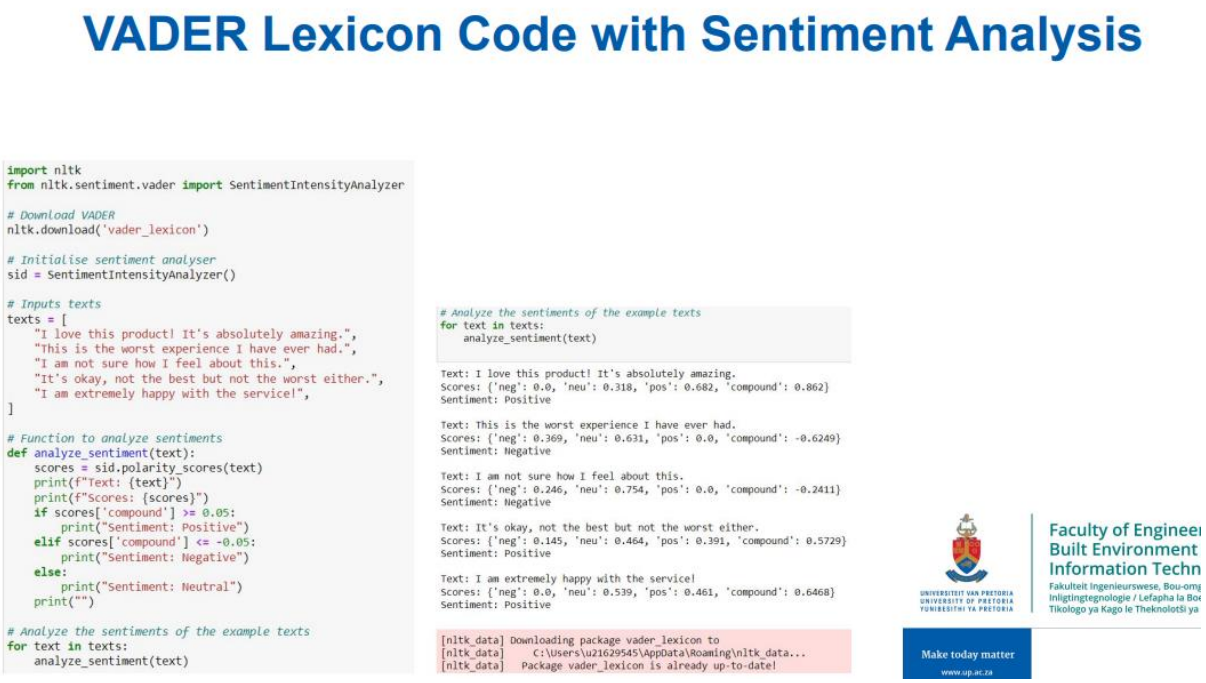
1. **Lexicon Coverage**:
   Coverage measures how well the lexicon includes the vocabulary used in a specific corpus or domain. A high-coverage lexicon should contain most of the words and expressions found in the target corpus, including rare or domain-specific terms. Evaluators often compare the lexicon to reference corpora or test sets to **calculate the percentage of words included.**
2. **Precision and Recall**:
   Like in machine learning, **precision** and **recall** are used to evaluate the quality of lexical entries. Precision measures the correctness of the entries (i.e., **how many of the words in the lexicon are accurate**), while recall measures how many of the actual words from the target dataset are included in the lexicon. **F1 score**, the harmonic mean of precision and recall, is often used for balanced evaluation.
3. **Sense Disambiguation and Accuracy**:
   For lexicons that include word senses or polysemy (multiple meanings of a word), it is crucial to ***evaluate how well the model assigns the correct meaning based on context***. Models are evaluated by **comparing their word sense assignments to a gold-standard dataset where human annotators have pre-identified the correct senses**. Tools like **Word Sense Disambiguation (WSD)** systems are commonly used to benchmark accuracy.
4. **Consistency and Redundancy**:
   Lexicons are evaluated for internal consistency, ensuring that similar words or concepts are treated similarly throughout the lexicon. Redundancy checks are also carried out to avoid duplication of entries or conflicting meanings, which can degrade the quality of NLP tasks like machine translation and sentiment analysis.
5. **Application-Based Evaluation**:
   A common method is to **test lexicon-based models within specific NLP tasks such as machine translation, sentiment analysis, or named entity recognition (NER)**. Evaluators measure the performance improvements provided by the lexicon compared to other models or against baseline methods without lexicons. For example, how well the lexicon enhances machine translation or sentiment analysis can be quantified by **BLEU scores** (for translation) or **accuracy** in sentiment classification.

Benchmarking against Standard Lexical Resources

Computational lexicons can be evaluated by comparing them with established resources such as WordNet, Oxford English Dictionary, or BabelNet. Metrics such as overlap in vocabulary, semantic relations, and coverage of word senses provide insights into the model's performance.

Computational lexicography models are evaluated both quantitatively (using statistical metrics) and qualitatively (through human evaluation and usability testing in NLP applications). This combined approach ensures that lexicons are both linguistically sound and practically useful in a wide range of AI-driven language tasks.

CODE VADER Lexicon Code with Sentiment Analysis



Useful links

1. https://mlu-explain.github.io/
2. https://github.com/23953005/final
3. https://github.com/firatolcum/Codecademy_Data_Analytics_Course/tree/master