Group 1

1. Dataset - https://snap.stanford.edu/data/gemsec-Deezer.html

The data was collected from the music streaming service Deezer (November 2017). These datasets represent friendship networks of users from 3 European countries. Nodes represent the users and edges are the mutual friendships. We reindexed the nodes in order to achieve a certain level of anonymity. The csv files contain the edges -- nodes are indexed from 0. The json files contain the genre preferences of users -- each key is a user id, the genres loved are given as lists. Genre notations are consistent across users. In each dataset users could like 84 distinct genres. Liked genre lists were compiled based on the liked song lists. The countries included are Romania, Croatia and Hungary. For each dataset we listed the number of nodes and edges.
Github: -https://github.com/benedekrozemberczki/GEMSEC

Task to be performed.

- Create interactive visualizations to enable users to explore, query, and interact with a given social network dataset.
- Provide functionalities for filtering, searching, zooming, and navigating through the network to facilitate exploration and analysis.
- Predict missing or potential connections between nodes in the social network.
- Identify influential nodes, opinion leaders, or key players in the social network based on centrality measures such as degree centrality, eigenvector centrality, or closeness centrality.

Group 2

2. **Spatial Data Visualization**

Dataset:-
https://www.kaggle.com/datasets/reubencpereira/spatial-data-repo This
data set is a collection of the following attributes :
Environmental Data: Vegetation indices, soil characteristics, evaporation
Climate Data: Temperature, precipitation, elevation
Socioeconomic and Demographic Data: Population Density, Access to major cities, Nightlight, Land usage
Conflict Data: Conflicts, death tolls, civilian casualties

Task to be performed.

- Perform exploratory data analysis and report its findings for each attribute of dataset.

- Discussion and insights on the spatial patterns, trends, and implications of satellite imagery to predict poverty for different locations.
- Perform exploratory data analysis to understand the characteristics and distributions of poverty among the listed country. Generate descriptive statistics, histograms, and scatter plots to visualize the spatial distributions for each attribute of the dataset.

Group 3

3. Multidimensional data visualization

Dataset:-
https://data.humdata.org/dataset/e2167f49-4ff7-4df7-82ca-bd223565e385/resource/487405b0-37fc-4d47-b3e7-8a034c6fc08d/download/multidimensional-poverty-index-trends.csv

- Apply various 3D visualization techniques to explore and analyze the poverty distribution using various attributes of the dataset.
- Visualize the feature Contribution of deprivation in dimension to overall poverty (% Living standards) with year and Multidimensional Poverty Index attributes.

Group 4

4. Hierarchical Data Visualization.

This type of visualization is commonly used to illustrate relationships, dependencies, and hierarchical structures within a dataset.

Dataset :- https://gist.github.com/mbostock/1044242#file-readme-flare-imports-json

Task to be performed:-

- Represent hierarchical relationships using nodes (vertices) and edges (lines connecting nodes). Each node represents a data element, and edges represent the hierarchical relationships between them.
- Design a sunburst chart to represent different clusters of the dataset.
- Use dataset features to represent attributes in the form of dendrogram visualization.
- Design Radial tree layouts to arrange hierarchical data in a circular or radial pattern, with nodes arranged around a central point.

Reference :- https://towardsdatascience.com/6-hierarchical-datavisualizations-98318851c7c5

Group 5

5. Map Data Visualization

Dataset:- https://data.world/getthedata/open-postcode-geo

Task to be performed

- Design a geo clustering data analysis based on various attributes.

- Plot points or markers on a map using latitude and longitude coordinates to represent specific locations or data points.
- Display spatial variations in each attribute using color-coded regions on a map.
- Plot paths, routes, or trajectories on a map using sequences of latitude and longitude coordinates.
- Develop interactive maps with zoom, pan, and hover-over capabilities for exploring spatial data.
- Overlay multiple layers of geographic data (e.g., points, lines, polygons) to analyze spatial relationships and intersections.

Group 6

6. Twitter data visualization(Static and Dynamic)

Dataset:- https://data.world/alexfilatov/2016-usa-presidential-election-tweets

Static dataset visualization using Python PySpark Library

- Apply topic modeling techniques (e.g., LDA) to identify key topics or themes discussed on Twitter within a given timeframe.
- Visualize topic clusters, co-occurrence networks, or word clouds to explore the most relevant topics and trending hashtags.
- Integrate twitter data with other sources such as news articles, stock prices, or weather data to analyze correlations and causal relationships.
- Design storytelling visualizations and narrative-driven analyses to communicate insights, trends, and key findings effectively to a non-technical audience.

Dynamic Twitter data visualizations

Note:- Use python stream library to generate streams of twitter dataset and highlight various trends and patterns.

Task need to performed:-

- Real time tweet trend analysis using key topics.
- Identify the key area where most candidates have focused during posting their tweets.
- Overall number of tweets recorded for each presidential election candidate.
- Categorize the topic of the tweets according to the interest of each candidate during the election.

Group 7

7. Facebook Network Analysis.

Dataset:- https://snap.stanford.edu/data/facebook_combined.txt.gz

Tasks need to be performed.

- Build a graph representation of the Facebook social network using the dataset. Each user corresponds to a node in the graph, and connections between users represent edges in the graph.
- Customize node and edge attributes (e.g., color, size, label) to encode additional information such as user attributes, centrality scores, or community memberships.
- Plot degree centrality histogram from the dataset.
- Plot distribution of shortest and longest path between edges.
- Plot Betweenness Centrality Histogram and closeness centrality histogram.
- Plot Clustering Coefficient Histogram of the dataset.
- Apply community detection algorithms (e.g., Louvain method, Girvan-Newman algorithm) to identify cohesive groups or communities of users within the Facebook network.
- Visualize detected communities as subgraphs or clusters within the larger network visualization.

Group 8

8. **Graph Data Analytics using Python PYSPARK,** GraphX, **NETWORKX and Visualization using other suitable library (NEO4j and BOKEH)**

Use Pyspark GraphX Library to determine following:

- How popular different users in the social network are (PageRank)
- Clusters of users based on how the users in the network are connected (Connected Components)
- Community detection and cohesiveness of the communities of users in the social network (Triangle Counting)

Use Case	**Dataset Source**	**Link**	**File Name**
PageRank	YouTube	https://snap.stanford.edu/data/com-Youtube.html	com-youtube.ungraph.txt
Connected Components	LiveJournal	https://snap.stanford.edu/data/com-LiveJournal.html	com-lj.ungraph.txt
Triangle Count	Facebook	https://snap.stanford.edu/data/egonets-Facebook.html	facebook_combined.txt

Group 9

9. **Graph Data Analytics using Python PYSPARK, NETWORKX and Visualization using other suitable library (NEO4j and BOKEH)**

Refer following Blog
https://databricks.com/blog/2016/03/16/on-time-flight-performance-with-graphframes-for-apache-spark.html

Following are the dataset details

- flights.csv - Dataset from Bureau of Transporation and Statistics. The dataset contains schedule and actual departure from major US airlines.(https://catalog.data.gov/dataset/airline-on-time-performance-and-causes-of-flight-delays-on-time-data)
- airports.dat - Dataset from openflights containing the list of airports including IATA code, airport name and airport location (https://openflights.org/data.html)

The example below uses the flight and airport data to illustrate some of the strengths of graph database. Provide a detailed step by step details for Data Capturing, Data Analysis and Data visualization techniques.Some of the questions like

- - Airport that is busiest
- - Which Airport causes the most delay
- - When do expect delay when flying from a location
- - Which flight are generally late when flying into a destination

In this example, while graph analysis is done using SPARK, graph visualization is achieved using a combination of BOKEH, NEO4J along with NETWORKX.

Group 10

10. Paradise-Panama-Papers Scam Analytics and Visualization

Dataset: https://www.kaggle.com/datasets/zusmani/paradisepanamapapers/data

Context

The Paradise Papers is a cache of some 13GB of data that contains 13.4 million confidential records of offshore investment by 120,000 people and companies in 19 tax jurisdictions (Tax Heavens - an awesome video to understand this); that was published by the International Consortium of Investigative Journalists (ICIJ) on November 5, 2017. Here is a brief video about the leak. The people include Queen Elizabeth II, the President of Columbia (Juan Manuel Santos), Former Prime Minister of Pakistan (Shaukat Aziz), U.S Secretary of Commerce (Wilbur Ross) and many more. According to an estimate by the Boston Consulting Group, the amount of money involved is around $10 trillion. The leak contains many famous companies, including Facebook, Apple, Uber, Nike, Walmart, Allianz, Siemens, McDonald's and Yahoo.

It also contains a lot of U. S President Donald Trump allies including Rax Tillerson, Wilbur Ross, Koch Brothers, Paul Singer, Sheldon Adelson, Stephen Schwarzman, Thomas Barrack and Steve Wynn etc. The complete list of Politicians involve is avaiable here.

The Panama Papers in the cache of 38GB of data from the national corporate registry of Bahamas. It contains world's top politicians and influential persons as head and director of offshore companies registered in Bahamas.

Offshore Leaks details 13,000 offshore accounts in a report.

With the help of Pyspark, and related visualization libraries , explore the following

1. How many companies and individuals are there in all of the leaks data
2. How many countries involved
3. Total money involved
4. What is the biggest best tax heaven
5. Can we compare the corruption with human development index and make an argument that would correlate corruption with bad conditions in that country
6. Who are the biggest cheaters and where they live
7. What role Fortune 500 companies play in this game

Group 11

11. Financial Analytics: Customer 360 Datasets

Dataset: https://www.kaggle.com/code/micjef/customer-360-degree-view/input

Dataset Description

The dataset has information of 100k orders from 2016 to 2018 made at multiple marketplaces in Brazil.

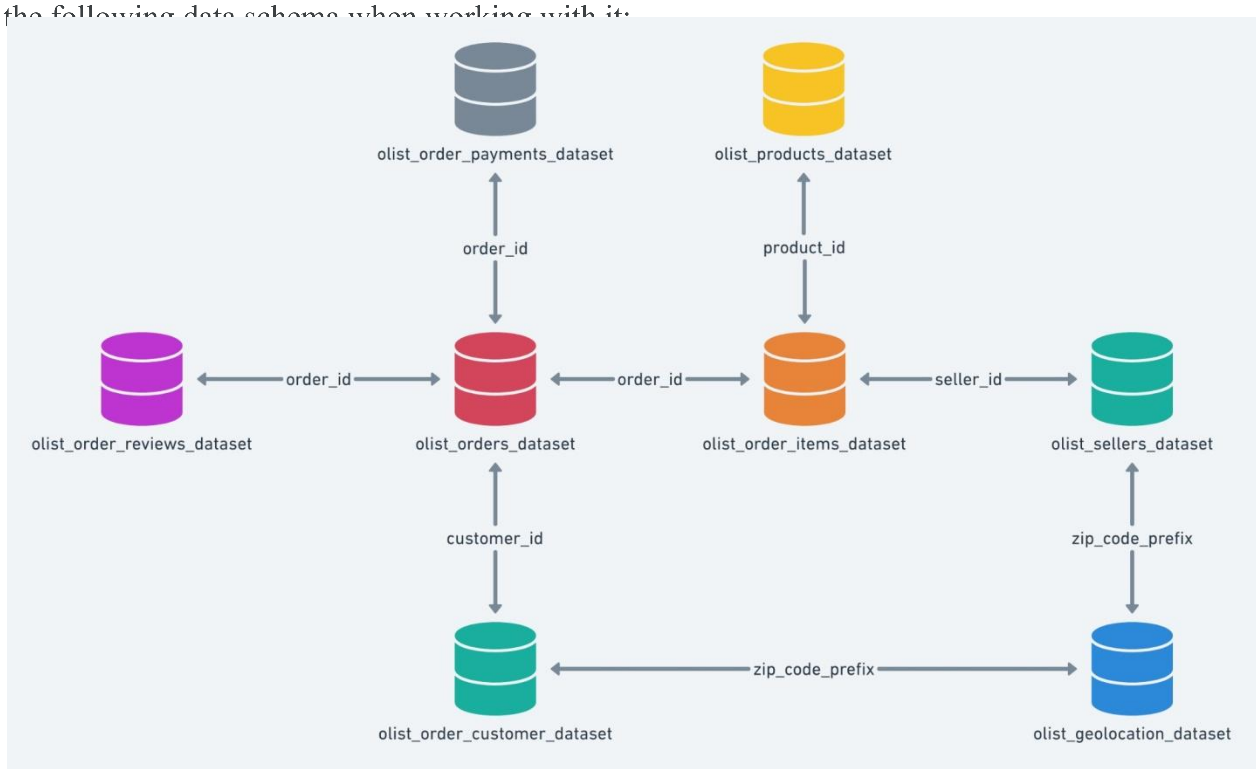
This is real commercial data, it has been anonymised.

This dataset was generously provided by Olist, the largest department store in Brazilian marketplaces. Olist connects small businesses from all over Brazil to channels without hassle and with a single contract. Those merchants are able to sell their products through the Olist Store and ship them directly to the customers using Olist logistics partners.

After a customer purchases the product from Olist Store a seller gets notified to fulfill that order. Once the customer receives the product, or the estimated delivery date is due, the customer gets a satisfaction survey by email where he can give a note for the purchase experience and write down some comments

Data Schema

The data is divided in multiple datasets for better understanding and organization. Please refer to the following data schema when working with it:



With the help of Pyspark, and related visualization libraries , explore the following

- Perfom basic EDA tasks for better understanding.
- Find the Top 10 products and sellers, analysis of orders by their geolocation and obtain information about Brazilian's online e-commerce profiles with the help of visualization libraries.
- Explore dataset to visualize
  - Product Profile
  - Customer Profle
  - Seller Profile
  - For example, Which product sells the most? Do they profit more from cheap or from expensive products?

    - Do orders vary throughout the year?
    - How much does the average customer spend? Does it vary with geographical location?
- Perform Seasonal Decomposition. It consists of three systematic components including level, trend, seasonality, and one non-systematic component called noise.
- How much does the average shop sell? Does it vary with geographical location?

Group 12

12. **Clickstream Analysis using Apache Spark and Apache Kafka**

Source: https://github.com/IBM/kafka-streaming-click-analysis

Clickstream analysis is the process of collecting, analyzing, and reporting about which web pages a user visits, and can offer useful information about the usage characteristics of a website.

Some popular use cases for clickstream analysis include:

- A/B Testing: Statistically study how users of a web site are affected by changes from version A to B.
- Recommendation generation on shopping portals: Click patterns of users of a shopping portal website, indicate how a user was influenced into buying something. This information can be used as a recommendation generation for future such patterns of clicks.
- Targeted advertisement: Similar to *recommendation generation*, but tracking user clicks "across websites" and using that information to target advertisement in real-time.
- Trending topics: Clickstream can be used to study or report trending topics in real time. For a particular time quantum, display top items that gets the highest number of user clicks.

USE Kafka, Pyspark and Data Visualization tools to present the useful insights.

Group 13

13. **Gephi based data visualization and analysis**

Dataset Link :-
https://www.kaggle.com/datasets/yessicatuteja/me-too-movement-tweets-dataset

Data Preparation and Import:

- Extract the provided ZIP file to access the dataset. Review the data to understand its structure and the types of relationships it contains (e.g., social interactions, citations, web links).
- Import the dataset into Gephi. Depending on the dataset format, you may need to use the 'Import Spreadsheet' feature for CSV files or the appropriate import option for other formats.

Network Graph Construction:

- Once the data is imported, Gephi will display the initial network graph. Use this as a starting point for your visualization project.

Basic Visualization Techniques:

- Layouts: Apply different layout algorithms to improve the graph's readability. Experiment with ForceAtlas2, Fruchterman Reingold, and Yifan Hu to see how each affects the visualization.
- Size and Color: Adjust the size and color of nodes and edges based on attributes like degree, centrality, or custom metrics present in your dataset.

Advanced Visualization and Analysis:

- Community Detection: Use the Modularity function to detect communities or clusters within the network. Color nodes based on their community to visually separate different groups.
- Centrality Measures: Calculate centrality measures (e.g., degree, betweenness, closeness) to identify important nodes. Highlight these nodes in your visualization.
- Filters: Apply filters to simplify the network, focusing on specific nodes or connections that are of interest.

Interpretation and Reporting:

- Interpret the visualized data. Identify any notable patterns, communities, key nodes, or other insights gained from your analysis.
- Prepare a report summarizing your findings. Include screenshots of your network visualization and descriptions of the techniques used and insights gained.

Group 14

14. **Gephi based data visualization and analysis**

Dataset Link :-
https://www.kaggle.com/datasets/noobidoobidoo/india-special-trains-dataset-2020-202 1

Dataset Import and Initial Exploration:

- Import the dataset into Gephi. Familiarize yourself with the initial network graph's appearance and explore basic dataset attributes.
- Document the network's basic properties (number of nodes, edges, etc.) using Gephi's Overview statistics.

Layout and Visualization:

- Apply a layout algorithm, such as ForceAtlas2, to improve the network's visual clarity. Experiment with layout settings to balance between aesthetic appeal and interpretability.
- Adjust node sizes and colors based on degree centrality to highlight more connected nodes.

Metric Calculation and Analysis:

- Calculate basic network metrics (degree, betweenness centrality) and intermediate metrics (eigenvector centrality, modularity for community detection).
- Identify and interpret the top nodes for each metric calculated. Discuss their potential role or significance within the network.

Community Detection and Visualization:

- Use the Modularity feature in Gephi to detect communities within the network. Adjust parameters to optimize community detection.
- Visualize communities by assigning unique colors to nodes based on their community membership. Analyze the number of communities, their sizes, and any notable characteristics.

Intermediate Network Dynamics:

- If the dataset includes temporal data, perform a basic dynamic analysis. Explore how the network evolves over time, focusing on changes in network size, community structure, or centrality measures.
- Visualize network dynamics using Gephi's Timeline feature, highlighting significant growth or shifts in the network.

Report :

- Compile a detailed report documenting the methodology, analysis, findings, and discussions. Include screenshots of the network visualizations and tables summarizing key metrics.

Group 15

Title - Design a Health Monitoring Dashboard using Elasticsearch and Kibana

Problem Statement

Develop a health monitoring system that leverages Elasticsearch and Kibana to visualize and track critical patient health parameters in real time. The system should use the Stroke Prediction Dataset to provide insights into stroke risk factors, enabling healthcare providers to monitor population health and identify high-risk individuals. The dashboard should represent each dataset feature graphically and support interactive exploration.

This project aligns with the research findings in:

“Smart healthcare monitoring system based on wireless sensor networks” (Springer, 2018), which emphasizes remote patient monitoring and decision support using real-time data visualization tools.

https://link.springer.com/article/10.1007/s11276-018-01896-2

Dataset

Source: Stroke Prediction Dataset – Kaggle Link:
https://www.kaggle.com/datasets/fedesoriano/stroke-prediction-dataset

Features:- gender- age- hypertension- heart_disease- ever_married- work_type- Residence_type- avg_glucose_level- bmi- smoking_status and - stroke (Target)

Project Objectives

1. **Data Ingestion into Elasticsearch:**

   - Preprocess the dataset and convert it to a format compatible with Elasticsearch (e.g., JSON or CSV).

   - Index the data using Logstash or Python Bulk API.

   - Map each feature correctly (categorical, numeric, boolean).

2. **Kibana Dashboard Design:**

   - Visualize each dataset feature using suitable charts:

     - Pie/Donut for gender, marital status

     - Histogram/Line for age, glucose levels

     - Bar/Stacked Bar for hypertension, heart disease, smoking status

- Heatmaps and alerts for stroke risk tracking

3. **Interactive Filtering & Drilldown:**

   - Add filters for gender, age, work_type, and stroke.
   - Enable dynamic drilldowns for subgroup analysis.

4. **Real-Time Health Monitoring Integration (Optional):**

   - Simulate streaming data using Filebeat or Logstash.
   - Display live data view and set alert triggers for high-risk values.

5. **System Architecture Overview:**

   - Data Source: Stroke Prediction CSV
   - Ingestion: Logstash / Python Bulk API
   - Search Engine: Elasticsearch
   - Visualization: Kibana Dashboard

Deliverables

1. Indexed Dataset: All health data features indexed into Elasticsearch
2. Kibana Dashboard: Visual dashboard with all feature-based panels and interactivity
3. Dashboard Screenshots: Snapshots of each visualization and filter
4. Technical Report: Description of setup, visualization choices, mappings, and insights some comparison with existing visualization, conclusion etc.
5. Source Code & Scripts: GitHub repo with ingestion scripts, mappings, and saved dashboards.

Optional Enhancements

- Deploy Kibana on a cloud platform (e.g., Elastic Cloud, AWS)
- Use Elastic Alerting to notify when stroke risk thresholds are breached

- Integrate with Machine Learning plugins in Elastic for anomaly detection

Group 16

Title - Investigating and Maintaining Sparse Decision Trees via Interactive Visualisation using TIMBERTREK

Problem Statement
Develop an interactive system using TIMBERTREK that aids in the exploration, maintenance, and evaluation of sparse decision trees for medical diagnosis. The system should leverage decision tree models built on the Non-Alcoholic Fatty Liver Disease (NAFLD) dataset to assist in understanding feature importance, pruning decisions, and model performance. The visual interface must enable real-time interaction with tree structures to improve interpretability and support decision-making.

This project aligns with the research findings in:
"Investigating and Maintaining Sparse Decision Trees via Interactive Visualisation" (IEEE, 2022), which highlights the use of visualization tools like TIMBERTREK for improved understanding and refinement of tree-based models.
Reference Link: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=9973202

Dataset

Source: Non-Alcoholic Fatty Liver Disease Dataset – Kaggle
Link: https://www.kaggle.com/datasets/utkarshx27/non-alcohol-fatty-liver-disease

Project Objectives

1. **Data Preparation:**
 - Clean and preprocess the dataset for modeling.
 - Handle missing values, encode categorical variables, normalize numerical features.

2. **Sparse Decision Tree Modelling:**
 - Train sparse decision trees using interpretable machine learning frameworks.
 - Apply pruning techniques to maintain model simplicity and interpretability.

3. TIMBERTREK Integration:
- Visualize the trained sparse decision trees using the TIMBERTREK interface.

- Allow interactive pruning, feature analysis, and node inspection.

4. **User-Centric Evaluation:**
   - Enable feedback-based interaction for refining tree models.
   - Provide support for exploring different thresholds and decision boundaries.

5. **Comparative Insights and Interpretability:**
   - Compare sparse trees vs full trees in terms of performance and usability.
   - Identify the most influential features contributing to NAFLD diagnosis.

6. **Visualization and Reporting:**
   - Present screenshots from TIMBERTREK interactions.
   - Highlight key visual patterns, model changes, and decision justifications.

Deliverables

1. Preprocessed Dataset: Cleaned and formatted NAFLD dataset.
2. Sparse Decision Trees: Trained interpretable models saved in compatible format.
3. TIMBERTREK Visualizations: Screenshots and exportable visual files from tree exploration.
4. Evaluation Report: Summary of insights from model interpretability and user interaction.
5. Source Code & Instructions: GitHub repo with modeling scripts, TIMBERTREK configs, and visualization steps.
6. Comparative Analysis: Summary of full vs sparse tree performance and interpretability findings.

Group 17

Title - Efficiency and Accessibility of Dengue Disease Through the Application of Data Visualization Dashboards Using Python

Reference Paper:
https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=9373291

Dataset: Dengue Fever Dataset

– Kaggle Link: https://www.kaggle.com/datasets/jyotidabas/dengue-fever-dataset

Problem Statement

Develop an interactive dashboard using Python for effective monitoring and visualization of dengue fever trends. This system aims to improve the efficiency and accessibility of dengue disease surveillance by representing critical factors such as symptoms, region, age group, and diagnosis visually. The dashboard will help public health officials and researchers to identify high-risk zones, track patterns, and support timely intervention.

Project Objectives

1. **Data Preprocessing and Transformation:**
   - Clean and normalize the Dengue dataset.
   - Handle missing values and encode categorical variables.
   - Aggregate data by location, age group, and time.
2. **Visualization Dashboard Development:**
   - Use Python libraries such as Plotly, Dash, and Streamlit.
   - Include charts for symptom distribution, diagnosis frequency, and region-wise patterns.
   - Heatmaps for geographical spread and time series plots for trends.
3. **Interactive Features and Filters:**
   - Filters for age, gender, symptoms, and diagnosis.
   - Drill-down for location-specific insights.
   - Real-time search for tracking clusters.
4. **Real-Time Simulation (Optional):**
   - Simulate dengue data using dummy stream feeds.
   - Enable dashboard updates using WebSocket or time intervals.

System Architecture Overview

1. Data Source: Dengue Fever Dataset (CSV)

2. Backend: Python (Pandas, NumPy)
3. Dashboard Framework: Dash / Streamlit / Plotly
4. Deployment: Localhost or cloud (Heroku, Streamlit Cloud)

Deliverables

1. Cleaned and processed dataset for visualization
2. Python-based interactive dashboard
3. Dashboard screenshots and summary insights
4. Technical report: design, code architecture, insights, comparisons
5. GitHub Repository with code and documentation

Group 18

Title - Implementing a Grafana Dashboard to Visualize Real-Time IoT Sensor Data for Anomaly Detection

Reference Paper:
"Real-time monitoring system using IoT-based occupancy detection and analytics for energy efficiency in buildings"
IEEE Link: https://ieeexplore.ieee.org/abstract/document/10287934

Dataset:
Occupancy Detection Dataset – Kaggle
Link: https://www.kaggle.com/datasets/saumitgp/occupancy-detection-dataset

Problem Statement

The goal of this project is to implement a real-time monitoring system that visualizes IoT sensor data using Grafana, enabling instant anomaly detection and actionable insights. The system should track key environmental parameters—temperature, humidity, and device status—across multiple locations. This setup supports energy-efficient building management, as proposed in the referenced research, by identifying unusual patterns or system failures in real-time.

Project Objectives

1. **R**eal-Time Data Ingestion:
    - Stream occupancy sensor data (temperature, humidity, light, CO_2, humidity ratio, and occupancy) using MQTT / Apache Kafka or simulated sources.
    - Store incoming data in Time-Series Database (TSDB) such as InfluxDB or Prometheus.
2. Grafana Dashboard Implementation:
    - Connect Grafana to the TSDB.
    - Create real-time visualizations for:
        - Temperature and humidity trends
        - Occupancy status by room
        - CO_2 levels and anomalies
        - Device online/offline status
3. Multi-Location Support:
    - Group and filter data based on building zones or room IDs.
    - Allow comparison between locations in terms of environmental parameters.
4. Anomaly Detection Integration:
    - Implement threshold-based alerting (e.g., temperature > 28°C or CO_2 > 1000 ppm).
    - Highlight abnormalities in Grafana panels using color codes or alert flags.
5. Alerts & Notifications:
    - Configure Grafana Alerting to send email/Slack notifications for out-of-range values or offline devices.

System Architecture Overview

- Sensors / Simulated Input: JSON/CSV stream of sensor values
- Data Ingestion Tool: Telegraf / MQTT Broker / Python Script
- Database: InfluxDB (time-series database)
- Visualization Tool: Grafana
- Optional Add-ons: Node-RED for simulation and control

Deliverables

1. Ingested Dataset: Real-time feed of occupancy sensor values stored in InfluxDB
2. Grafana Dashboard:
    - Line charts for temperature, humidity, CO_2
    - Gauge/Status panels for occupancy and device status
    - Alert panels for abnormal readings
3. Alert Configuration: Email/Slack alerts for high risk values or system malfunctions
4. Screenshots and Report:
    - Dashboard views
    - Annotated anomaly examples
    - Insights derived from visualizations
5. Source Code: GitHub repository with data simulator, ingestion scripts, and dashboard export
6. Technical Documentation:
    - Description of architecture
    - Dashboard panels setup

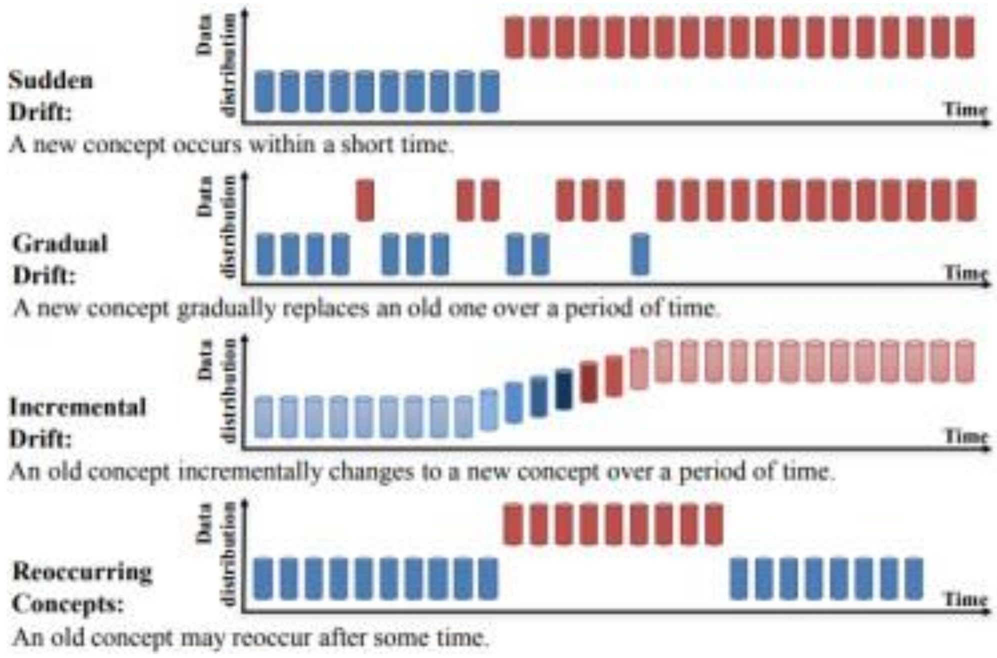
    - Thresholds used for alerts
    - Comparison with paper findings

Concept Drift Datasets

Concept drift, also known as model drift, occurs when the task that the model was designed to perform changes over time. For example, imagine that a machine learning model was trained to detect spam emails based on the content of the email. If the types of spam emails that people receive change significantly, the model may no longer be able to accurately detect spam.

Concept Drift can be further divided into four categories (Learning under Concept Drift: A Review, Jie Lu et al.):

- Sudden Drift
- Gradual Drift
- Incremental Drift
- Recurring Concepts


Sudden Drift:
A new concept occurs within a short time.
Gradual Drift:
A new concept gradually replaces an old one over a period of time.
Incremental Drift:
An old concept incrementally changes to a new concept over a period of time.
Reoccurring Concepts:
An old concept may reoccur after some time.
Data distribution
Time

Use the following datasets and describe the Drift Type present in these datasets. Use effective visualization techniques to highlight the area where drift is occurring.

Based on above

Group 19

SEA Concepts

Source: https://github.com/vlosing/driftDatasets/tree/master/artificial/sea

This dataset consists of 50000 instances with three attributes of which only two are relevant. The two class decision boundary is given by $f_1 + f_2 = b$, where f_1, f_2 are the two relevant features and b a predefined threshold. Abrupt drift is simulated with four different concepts, by changing the value of b every 12500 samples. Also included are 10% of noise.

Weather (original source)

Source: https://github.com/vlosing/driftDatasets/tree/master/realWorld/weather Elwell et al. introduced this dataset. In the period of 1949-1999 eight different features such as temperature, pressure, wind speed etc. were measured at the Offutt Air Force Base in Bellevue, Nebraska. The target is to predict whether it is going to rain on a certain day or not. The dataset contains 18159 instances with an imbalance towards no rain (69%).

Group 20

Rotating Hyperplane

Source:https://github.com/vlosing/driftDatasets/tree/master/artificial/hyperplane

A hyperplane in d-dimensional space is continuously changed in position and orientation continuous addition. We used the Random Hyperplane generator in MOA with the same parametrization as in PAW (10 dimensions, 2 classes, delta=0.001).

Electricity market dataset (original source)

Source:https://github.com/vlosing/driftDatasets/tree/master/realWorld/Elec2

This problem is often used as a benchmark for concept drift classification. Initially described by Harris et al. it was used thereafter for several performance comparisons. A critical note to its suitability as a benchmark can be found in. The dataset holds information of the Australian New South Wales Electricity Market, whose prices are affected by supply and demand. Each sample, characterized by attributes such as day of week, time stamp, market demand etc., refers to a period of 30 minutes and the class label identifies the relative change (higher or lower) compared to the last 24 hours. We used the normalized version as it also can be found here.

Group 21

Moving RBF

Source:https://github.com/vlosing/driftDatasets/tree/master/artificial/rbf
Gaussian distributions with random initial positions, weights and standard deviations are moved with constant speed v in d-dimensional space. The weight controls the partitioning of the examples among the Gaussians. We used the Random RBF generator in MOA with the same parametrization as in PAW (10 dimensions, 50 Gaussians, 5 classes, v=0.001).

Forest Cover Type (original source)

Source: https://github.com/vlosing/driftDatasets/tree/master/realWorld/covType Assigns cartographic variables such as elevation, slope, soil type, ... of 30 x 30 meter cells to different forest cover types. Only forests with minimal human-caused disturbances were used, so that resulting forest cover types are more a result of ecological processes. It is often used as a benchmark for drift algorithms. We used the normalized version as it also can be found here.

Group 22

Interchanging RBF **(Own dataset: When used in publication please cite http://ieeexplore.ieee.org/document/7837853/)**

Source:https://github.com/vlosing/driftDatasets/tree/master/artificial/rbf
Fifteen Gaussians with random covariance matrices are replacing each other every 3000 samples. Thereby, the number of Gaussians switching their position increases each time by one until all are simultaneously changing their location. This allows to evaluate an algorithm in the context of abrupt drift with increasing strength. Altogether 67 abrupt drifts are occurring within this dataset.

Poker Hand (original source)

Source: https://github.com/vlosing/driftDatasets/tree/master/realWorld/poker One million randomly drawn poker hands are represented by five cards each encoded with its suit and rank. The class is the resulting poker hand itself such as one pair, full house and so forth. This dataset has in its original form no drift, since the poker hand definitions do not change and the instances are randomly generated. However, we used the version presented in PAW, in which virtual drift is introduced via sorting the instances by rank and suit. Duplicate hands were also removed. We used the normalized version as it also can be found here.

Group 23

Moving Squares **(Own dataset: When used in publication please cite http://ieeexplore.ieee.org/document/7837853/)**

Source: https://github.com/vlosing/driftDatasets/tree/master/artificial/movingSquares
Four equidistantly separated, squared uniform distributions are moving in horizontal direction with constant speed. The direction is inverted whenever the leading square reaches a predefined boundary. Each square represents a different class. The added value of this dataset is the predefined time horizon of 120 examples before old instances

may start to overlap current ones. This is especially useful for dynamic sliding window approaches, allowing to test whether the size is adjusted accordingly.

Outdoor Objects (Own dataset: When used in publication please cite http://ieeexplore.ieee.org/document/7280610/)

Source: https://github.com/vlosing/driftDatasets/tree/master/realWorld/outdoor We obtained this dataset from images recorded by a mobile in a garden environment. The task is to classify 40 different objects, each approached ten times under varying lighting conditions affecting the color based representation. Each approach consists of 10 images and is represented in temporal order within the dataset. The objects are encoded in a normalized 21-dimensional RG-Chromaticity histogram.

Group 24

Transient Chessboard (Own dataset: When used in publication please cite http://ieeexplore.ieee.org/document/7837853/)

Source: https://github.com/vlosing/driftDatasets/tree/master/artificial/chess Virtual drift is generated by revealing successively parts of a chessboard. This is done square by square randomly chosen from the whole chessboard such that each square represents an own concept. Every time after four fields have been revealed, samples covering the whole chessboard are presented. This reoccurring alternation penalizes algorithms tending to discard former concepts. To reduce the impact of classification by chance we used eight classes instead of two.

Mixed Drift (Own dataset: When used in publication please cite http://ieeexplore.ieee.org/document/7837853/)

Source: https://github.com/vlosing/driftDatasets/tree/master/artificial/mixedDrift The datasets Interchanging RBF, Moving Squares and Transient Chessboard are arranged next to each other and samples of these are alternately introduced. Therefore, incremental, abrupt and virtual drift are occurring at the same time, requiring a local adaptation to different drift types.

Group 25

Rialto Bridge Timelapse (Own dataset: When used in publication please cite http://ieeexplore.ieee.org/document/7837853/)

Source: https://github.com/vlosing/driftDatasets/tree/master/realWorld/rialto Ten of the colorful buildings next to the famous Rialto bridge in Venice are encoded in a normalized 27-dimensional RGB histogram. We obtained the images from time-lapse videos captured by a webcam with fixed position. The recordings cover 20 consecutive days during may-june 2016. Continuously changing weather and lighting conditions affect the representation. We generated the labels by manually