Datasets

- **OpenAQ**: real-time and historical air quality data from over 100 countries.
- **GEMS/Water Data**: global water quality data.
- **European Soil Data Centre (ESDAC)**: soil contamination.
- **NASA EarthData**: global climate indicators.
- **Global Biodiversity Information Facility (GBIF)**: occurrence records of species collected from field observations, museum specimens, and citizen science platforms.
- **Global Land Cover (GLC) Dataset**: global vegetation index from satellite imagery.
- **Landsat Data (USGS)**: global land cover dataset with multiple classes.
- **xBD: A Dataset for Assessing Building Damage from Satellite Imagery**: satellite imagery containing annotations for building damage.
-

1: Environment & Pollution

Environment

- **Environment**: complex system that consist of biotic environemnt (living organisms) and abiotic environment (not-living physical surroundings that support life).
- Abiotic environment: **atmosphere**: layer of gases surrounding Earth; **hydrosphere**: all water bodies; **lithosphere**: solid outer part of the Earth.
- **Biosphere**: system encompassing both biotic and abiotic components.
- Boundary between natural conditions and human-induced environmental issues is often blurred, as changes in natural conditions can make worse human-caused environmental problems, and human-induced issues can significantly alter natural conditions (e.g., global warming).

Polution

Pollution: alteration of physical, chemical or biological characteristics of soil, water or air that negatively impacts ecosystems, human health or natural processes.

1. **Physical pollution**: harmful or undesirable changes in the physical characteristics of soil, water, or air. E.g., **thermal** (industrial processes release heated water), **turbidity in water** (water, sediment increases the cloudiness of water, reducing sunlight penetration), **particulate matter (PM)** (air, dust, smoke reduces air quality), **physical pollutans** (soil, plastic debris affecting water absorption, nutrient availability).
2. **Chemical pollution**: introduction of harmful or toxic chemicals into the environment altering the chemical balance of soil, water or air. **Heavy metals in water and soil** (lead, mercury poison living organisms), **nitrogen oxides and sulfur oxides in air** (vehicle emissions and industrial processes), **pesticides in water and soil** (agriculture, contaminate the environment and harm wildfile, aquatic life, human health).
    1. **Bioaccumulation**: process by which toxic substances build up in the tissues of organisms over time at higher concentrations than in the surrounding environment.
    2. **Biomagnification**: process by which the concentration of toxic substances increases at each successive level of a food chain, leading to higher toxicity in top predators.
3. **Biological pollution**: introduction of harmful or invasive living organisms into the environment. **Microbial contamination of water** (harmful bacteria, viruses, or parasites, often from sewage, agricultural runoff, or untreated wastewater), **invasive species** (non-native organisms introduced to an ecosystem,and outcompete native species for resources, disrupt food chains), **pathogenic microbes in soil** (spread through agricultural practices or contaminated water, affecting plant health, crop yields, and soil fertility).
    1. **Pathogen**: microorganism, such as a virus, bacterium, or fungus, that causes disease in its host. Pathogens in water are typically quantified using **E. coli**: type of bacteria commonly found in the intestines of humans and animals. Its presence suggest fecal contamination.

Quantifying Pollution

- Pollution in water, air, or soil is quantified either directly by measuring the **concentration** of the pollutant — defined as the amount of a substance present in a specific volume or mass of water, air, or soil (e.g., mg/L for water) — or through **indirect indicators**, which are proxy measurements suggesting the presence of pollutants, such as **biological oxygen demand (BOD)**, which measures the amount of oxygen consumed by microorganisms during the decomposition of organic matter (high $\rightarrow$ organic pollution), or **chlorophyll levels in water** or **algae blooms**.

- Common units are **mg/L**, μ**/L**, **ppm** (parts per **million**, number of pollutant molecules per million molecules of air or water), **ppb** (parts per **billion**), **g/kg**.
- **Nutrient pollution**: excessive introduction of nutrients, particularly nitrogen and phosphorus, into water bodies from sources like agricultural runoff and sewage. Imbalance overwhelms ecosystems, causing e.g., algae bloom.
  - **Nutrient**: substance that provides essential elements for the growth and metabolism of living organisms, such as nitrogen, phosphorus, and potassium.
  - **Algae bloom**: rapid increase in the population of algae in water bodies.

Composite Indices

- **Composite indices**: combine the effects of multiple pollutants, providing a more comprehensive assessment of pollution levels.
- **Air quality index (AQI)**: combines the concentrations of multiple air pollutants, including particulate matter (PM2.5 and PM10), ozone (O3), sulfur dioxide (SO2), nitrogen dioxide (NO2), and carbon monoxide (CO).

Sources of Pollution in Soil and Water

- **Water**:
  1. **Point sources**: specific, identifiable locations where pollutants are discharged directly into water bodies or soil (e.g., industrial facilities, oil spills).
  2. **Non-point sources**: diffuse, widespread sources of pollution that are harder to trace to a single discharge point, often carried by runoff from the land (e.g., urban runoff).
- **Air**:
  1. **Mobile sources**: move from place to place, primarily associated with transportation (vehicles, aircraft).
  2. **Stationary sources**: non-moving sources that release pollutants from a specific location or facility (e.g., power plants).
- **Volatile organic compounds (VOCs)** are a group of organic chemicals that easily evaporate into the air, contributing to air pollution and smog formation.

The Source-Pathway-Receptor Model

Source-pathway-receptor model is a framework used in environmental science to understand and assess how pollutants **move through the environment** and affect living organisms or ecosystems. It **breaking down the interactions into understandable units**, **helps in pinpointing the critical stages where interventions can be applied**, **allows for accurate assessments of health risks, environmental damage, and the overall safety of a population**. By analyzing exposure data (e.g., pollutant concentrations), decision-makers can craft tailored policies. **Exposure data** is crucial for building models that **predict the future impacts of pollution**.

- **Source**: origin of pollution.
- **Pathway**: route (e.g., air, water, soil) through which the pollutant travels from the source to reach the environment and potentially the receptor.
- **Receptor**: organism, population, or ecosystem that is affected by the pollutant (e.g., humans, wildlife).
- **Exposure**: contact between the pollutant and the receptor.

Questions addressed by EDA

1. Pollution source identification.
2. Pollution prediction and forecasting.
3. Pattern and anomaly detection.
4. Impact assessment and risk prediction.
5. Optimization of mitigation strategies.
6. Spatio-semporal modeling.
7. Causal inference and relationships.
8. Climate pollution interaction.

2: Climate & Hydrologic Cycle

- **Climate**: long-term pattern of weather conditions in a particular region, typically averaged over a period of 30 years or more. Variables are **averaged over long periods**.
- **Weather**: short-term atmospheric conditions, including temperature, precipitation, and wind, that occur in a specific place over a brief period, such as hours or days. Variables are **measured in real-time or over short periods**.

Variables

- **Wind speed**: rate at which air moves horizontally across the Earth's surface (m/s).
- **Wind direction**: direction from which the wind is blowing, typically measured in degrees from true north (0 to 360). **Wind rose** is used to plot wind speed and direction.
- **Temperature**: measure of the thermal energy or warmth of the air in the atmosphere. Measured in **shade** at **2 meters** above groundm to avoid solar radiation and surface influence. **Thermometer**.
- **Precipitation**: any form of water, such as rain, snow, sleet, or hail, that falls from the atmosphere to the Earth's surface. Measured **regardless of sunlight** using devices that collect and quantify the amount of water reaching the surface. **Rain gauge**.
- **Humidity**: amount of water vapor present in the air, percentage of the maximum amount the air can hold at a given temperature (relative humidity). **Hygrometer**,
- **Atmospheric pressure**: force exerted by the weight of the Earth's atmosphere on a specific point. Low pressure $\rightarrow$ stormy weather (air rises because the weight of the atmosphere above is lower, then it cools and expands, leading to condensation and the formation of clouds), high pressure $\rightarrow$ clear conditions. **Barometer**.
- **Solar radiation**: energy emitted by the Sun that reaches the Earth's surface in the form of electromagnetic waves, primarily visible light, ultraviolet (UV), and infrared (IR). **Radiometer**.
- **Weather station**: station equipped with sensors, which continuously monitor environmental conditions. Data entered **manually** or **automatically**.
- **Weather (doppler) radar**: instrument that use radio waves to detect and track precipitation (**location, intensity, and movement**) by bouncing radio signals off of particles in the atmosphere.

Weather and Climate Models

- **Weather model**: mathematical model used to simulate and predict **short**-term atmospheric conditions.
- **Climate model**: mathematical model used to simulate and predict **long**-term atmospheric conditions.
- **Reanalysis**: procecss of reconstructing past weather and climate conditions.

Hydrologic Cycle

- **Hydrologic (water) cycle**: continuous process by which water moves through the Earth's atmosphere, surface, and subsurface, driven by **evaporation, condensation, precipitation, infiltration, and runoff**.
  - **Evaporation**: rate at which water is converted from liquid to vapor on the surface of the earth,
  - **Transpiration**: amount of water released by plants into the atmosphere through small openings in their leaves.
  - **Infiltration rate**: speed at which water permeates the soil.
  - **Runoff**: water from precipitation, snowmelt, or other sources that flows over the land surface and moves toward rivers, lakes, or oceans (**overland flow** and **shallow subsurface flow**).
  - **River discharge**: volume of water flowing in rivers or streams.
  - **Groundwater recharge**: volume of water replenishing underground aquifers.
  - **Soil moisture**: The amount of water held in the soil.

Water Resources Management

- **Water resources management (WRM)**: process of planning and implementing strategies for the optimal extraction, use, conservation, and recycling of water to meet both human and environmental needs, while ensuring sustainability and minimizing water scarcity, pollution, and environmental degradation.
  - **Water consumption**: amount of water used by different sectors/users in agriculture, industry, households, and municipalities.
  - **Water availability**: volume of water available from sources such as rivers, lakes, reservoirs, and aquifers.
    - **Water demand**: projected need for water in various sectors (agricultural, industrial, domestic) based on population growth, economic development, and climate patterns.
    - **Water quality**: indicators of water health, including parameters such as pH, dissolved oxygen, nutrient levels, and pollutants.
    - **Groundwater levels**: height of water in aquifers.
    - **Reservoir storage levels**: volume of water stored in artificial or natural reservoirs.
    - **Irrigation efficiency**: percentage of water used effectively in agricultural irrigation systems.

Solar, Wind and Hydro Energy

- **Renewable energy**: energy derived from natural sources that are **continuously replenished**, such as sunlight, wind, and water. They **combat climate change** due to **sustainability**, **low environmental impact and greenhouse**

gas emission and being **cost-effective** and **reliable**. (EU **40%**).
  - **Solar energy**: captured sunlight using solar panels or solar thermal systems and converted into electricity or heat. **Solar energy potential** is the total amount of solar energy that can be harnessed over a time period.
  - **Wind energy**: converted kinetic energy of moving air into electricity using wind turbines. **Wind power density** is the amount of power available from the wind (derived from wind speed and air density).
  - **Hydro energy**: harnessed energy of flowing water, typically from rivers or reservoirs. WRM plays a critical role in regulating reservoir levels to ensure a steady supply of water for hydroelectric generation.

Climate Change

- **Climate change**: **long-term alteration** of climate conditions on Earth, primarily driven by increasing concentrations of greenhouse gases, such as carbon dioxide, methane, and nitrous oxide, in the atmosphere, which trap heat in the Earth's atmosphere causing global temperatures to rise (**global warming**). Greenhouse gases allow sunlight in but prevent some of the heat from escaping. **Rising global temperatures**, **sea level rise**, **altered precipitation patterns**, **ocean acidification**, **increased frequency of extreme weather events**. Climate change is supported by **empirical evidence** through: **rising global temperatures**, **ice core data** (atmospheric CO2 levels in ice, like a time capsule, showing levels have remained relatively stable until the industrial revolution), **satellite observations** (shrinking ice caps, melting glaciers), **ocean warming and acidification**.

3: Biodiversity, Agriculture, Land Use

Ecology and Biodiversity Data

- **Biodiversity**: variability among living organisms from all sources, including terrestrial, marine, and aquatic ecosystems, as well as the ecological complexes they are part of.
  - **Species**: group of organisms capable of interbreeding and producing fertile offspring, sharing common characteristics and classified under the same biological category.

Data:

1. **Single species occurrence**: **species observation records** (point-in-time data detailing where and when a specific species was observed).
2. **Single species movement**: **tracking and migration patterns**, **home range and territory use**.
3. **Population-level**: **population density and distribution**, **demographic data** (population dynamics, e.g., birth rates, death rates).

- **Population-level**: study of a group of individuals of the same species living in a specific geographic area.
- **Transect surveys**: surveys where researchers walk or fly along fixed paths (transects) and record the number of individuals encountered.

4. **Community-level**: **Species richness and diversity metrics**, **ecological interaction networks** (interactions among species).

- **Community-level**: analysis of multiple interacting species living within a shared environment.
- **Quadrat sampling**: researches mark put small, square plots (quadrats) in the study area and count how many different species and individuals are found within the quadrat.

5. Data is usually captured using **camera traps, tracking devices like animal tags, or sampling from feces**.

Sustainable Agriculture

- **Sustainable agriculture**: farming practices that meet current food and textile needs without compromising the ability of future generations to meet their own needs. Minimize environmental impact by promoting biodiversity, improving soil health, conserving water, reducing the use of chemical inputs.
- **Precisio agriculture**: **data-driven farming approach** that uses advanced technologies such as GPS, remote sensing, and IoT sensors to monitor and manage variability in crops, soil, and environmental conditions, **optimizing resource use** and **maximizing productivity** at a granular, site-specific level.

Data:

1. **Crop data**: crop health index (images), crop yield (time-series, spatial).
2. **Soil data**: soil moisture (time series, map).
3. **Land use and land cover data** (image, map).
4. **Water usage data** (time series).

Questions answered by EDA: **crop health monitoring**, **water management optimization**, **soil quality assessment**, **climate resilience**, **sustainability and carbon footprint**.

Land Use and Land Cover

- **Land use**: human activities or economic functions associated with a particular piece of land, such as agriculture, urban development.
- **Land cover**: physical material on the Earth's surface, such as vegetation, water bodies, or artificial structures like roads and buildings.
- **Labelled data** is generated using:
  - **Ground truthing**: field surveys and observations are conducted to manually classify specific areas
  - **Expert interpretation**: experts manually interpret high-resolution satellite images or aerial photographs to label land cover types
  - **Existing maps and datasets**: previously validated land use or land cover maps can be used as labeled datasets for training models on similar regions.
  - **Crowdsourcing**: platforms like Google Earth Engine enable public participation in labeling imagery.
- **Maching learning tasks**:
  - **Change detection**: analyzing satellite imagery or remote sensing data over different time periods to detect and quantify changes in land use or land cover.
  - **Prediction or forecasting**: forecast future land use changes based on historical data and influencing factors such as population growth, economic development.
  - **Anomaly detection**: anomalies or outliers in land use patterns, such as illegal logging or unexpected changes in natural habitats.
  - **Land use optimization**: optimize land use patterns for environmental, economic, or social benefits, such as determining best locations for conservation efforts.

Disaster Management

- **Risk assessment and mapping**: identify areas most vulnerable to disasters and create risk maps.
- **Early warning systems**: develop predictive models for natural disasters.
- **Real-Time Monitoring**: provide situational awareness during disasters.
- **Damage assessment**: assess the extent of damage in affected areas.

4: Characteristics of Environmental Data

Types of Environmental Data

1. **In-situ**: refers to data collected directly at location of interest (person or automated). Usually has high accuracy, high spatial resolution data, allows to monitor short-term variations (high-frequency) and trends over time, provide depth or vertical profiles (e.g., soil layers), often includes contextual information such as weather conditions. Includes **time-series data, scalar values, text, images**.
   1. **High-frequency**: continuously recorded data, often at small intervals of seconds, minutes, etc. Can be stored and retrieved in batches (offline data) or transmitted in real-time (online data). E.g., weather stations, air quality monitors.
   2. **Low-frequency**:
      1. **Periodic**: periodic sampling at regular larger intevals. E.g., weather quality sampling, biodiversity surveys.
      2. **Event-based**: triggered by specific events, such as storms. E.g., storm weather sampling, post-wildfire surveys.
   3. **Crowdsource**: collected by ordinary citizens, often referred to as citizen scientists. Use personal devices, like smartphones, to collect data. This method needs to be guided and well-organized. E.g., **CrowdWater** app to record water levels and stream conditions.
2. **Remote sensing (RS)**: acquisition of information about the Earth's surface and atmosphere from a distance, typically using platform such as satellites, aircrafts, drones, balloons. Generally **low-frequency** and **periodic** (satellites), or **event-based** (drones). It is common for RS data to be **calibrated using in-situ observation data** to improve accuracy. Includes **2D images, 3D point clouds**.
3. **Model-generated**: produced by computational models that simulate environmental processes, such as climate models. Can provide **high- and low-frequency** data. Common to be **calibrated using in-situ observation data**. Includes **time-series, 2D maps (images, gridded data), 3D voxel or point cloud data, scalar values**.
4. **Narrative**: mostly **textual or qualitative** data collected from written or spoken sources that provide **context**, **descriptions**, and **insights** to **complement quantitative data** from in-situ, remote sensing (RS), or model-generated

sources. Involves **text mining**, **web scraping**, or **document scraping**. Includes **mostly text as scalar values or audio**.

Characteristics of Environmental Data

1. **Spatial nature**: location-specific data.
2. **Temporal nature**: environmental processes evolve over time.
3. **Seasonality and cyclic patterns**: exhibits seasonal trends.
4. **Spatial and temporal correlation**: nearby locations or time points tend to be correlated.
5. **Non-linearity**: non-linear relationships between variables, i.e. small changes in one variable can lead to disproportional changes in another.
6. **Scale dependency**: patterns or relationships that are evident at one scale (local, regional, global) may not be visible or behave differently at another scale.
7. **Multi-modality**: collected in different forms or modalities (images, time-series, etc.)
8. **Uncertainty**: discrepancies between the quantities that describe a system (such as measurements or model predictions) and the actual state of the system. E.g., measurement uncertainity, representativity uncertainity, interpolation uncertainty.
9. **Heterogeneity**: scale, resolution, and uncertainty of different subsets of the data can vary significantly.
10. **Large volume**: massive datasets.
11. **Non-normality**: does not follow normal distribution.
12. **Missing data**: missing data due to sensor failures, data collection issues.

Metadata

Metadata refers to information that **describes and provides context** about the actual data, such as its source, collection methods, accuracy, resolution, and format. Given that environmental data is **heterogeneous and scale-dependent**, integrating and analyzing it without detailed metadata would be extremely challenging, if not impossible. Environmental data inherently carries **uncertainty**, and without detailed metadata, such as **measurement precision, sensor accuracy, model assumptions, and calibration details**, uncertainty quantification and proper error analysis cannot be effectively performed, compromising the reliability and validity of the analysis.

1. **In-situ**: location (latitude, longitude, and sometimes altitude), time of collection, instrument type and settings, calibration details.
2. **Remote sensing**: satellite or sensor type and orbit, spatial resolution (e.g., pixel size), temporal resolution (e.g., revisit frequency), spectral bands used, processing level (raw, processed, etc.), georeferencing information (coordinate system, projection).
3. **Model-generated**: model type and version, input data sources, model assumptions and parameters, spatial and temporal resolution, uncertainty estimates, calibration and validation methods.
4. **Narrative**: source of the text or audio (e.g., survey, report, social media), date of collection or publication, author or contributor details, method of data extraction (e.g., text mining, web scraping), language and format, any categorization or tags related to the content.

5 & 6: Geospatial Data

Geospatial data is information that has geographic aspect. By associating environmental data with specific geographic coordinates, geo-spatial data facilitates the **analysis of spatial relationships and patterns**. Also, by mapping different data to same spatial coordinates, it enables the **integration of diverse environmental datasets**.

Geospatial Reference

- **Coordinates**: **coordinate system** is essential, as same numerial coordinates can point to different locations depending on the system used.
- **Addresses**: don't require a coordinate system, instead refer directly to a specific location, which can be converted into coordinates using **geocoding**: a process of convering textual geographical identifies into geographic coordinates.
- **Other geographical identifiers**: postal codes, place names, administrative boundaries.

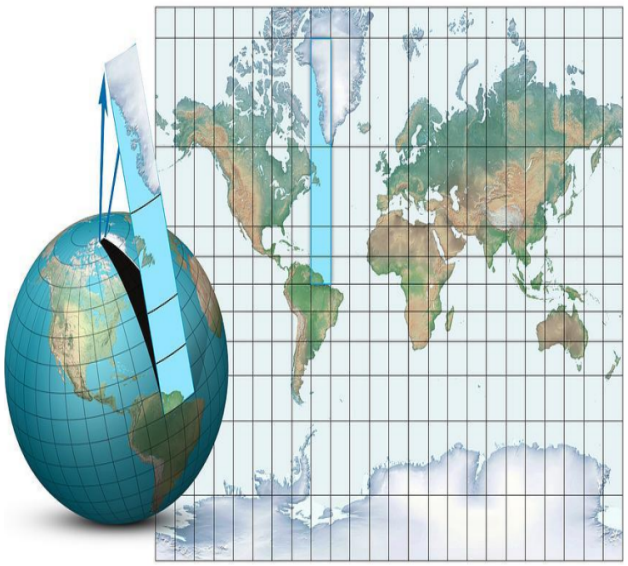
Geo-spatial Coordinates

1. **Latitude and longitude**: spherical coordinate system using angular measurements and based on the **Equator** and **Prime Meridian** as reference lines. **Latitude** is **polar angle** (how far north or south a point is from the Equator,

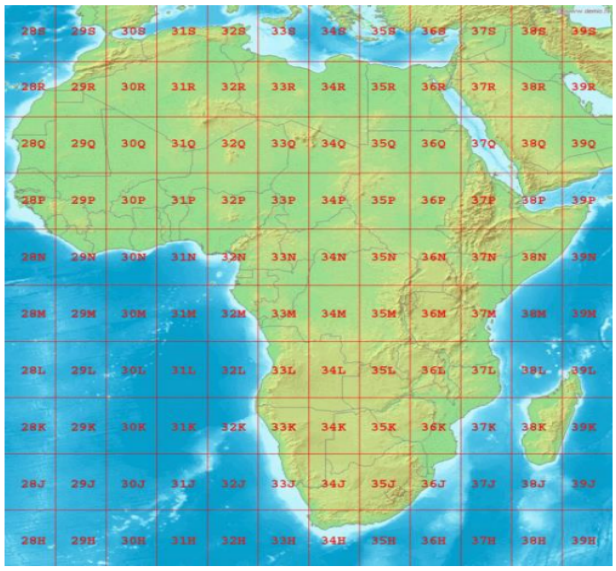
which is 0 is latitude) with values ranging from -90 at South Pole to +90 at North Pole. **Longitude** is **azimuthal angle** (how far east of west a point is from the Prime Meridian, which is at 0 longitude), with values ranging from -180 to 180+.

- **Spherical coordinate system**: A three-dimensional coordinate system where a point's position is a coordinate system defined by three values: the **radial distance from a central point** (often called the origin), the **polar angle** (the angle measured from a reference axis, usually the z-axis), and the **azimuthal angle** (the angle measured in the plane perpendicular to the reference axis, typically from a reference direction in that plane, such as the x-axis).
- **Equator**: The imaginary line around the middle of the Earth, **equidistant from the North and South Poles**, dividing the Earth into the Northern and Southern Hemispheres.
- **Prime Meridian**: The imaginary line running from the North Pole to the South Pole through Greenwich, England. It divides the Earth into the Eastern and Western Hemispheres.
- **WGS84 system**: ensures the accuracy of lat/long locations descriptions by defining the Earth's shape and reference points. It provides a mathematical model (an ellipsoid) that closely approximates this shape. It extends the system to include elevation, enabling 3D positioning.

2. **Universal transverse Mercator (UTM)**: **planar coordinate system** used to describe locations on Earth's surface by dividing it into a series of zones, each of which is mapped onto a flat grid using transverse Mercator projection. UTM system projects the Earth's curved sufrace onto a series of 2D flat grids. This is done by dividing the globe into 60 longitudinal zones, each covering 6 degrees of longitude. Then apply **transverse Mercator projection** to each zone, which maps cylidrical surface onto a plane. Each zone has a unique grid. A location's position is described by **easting** (horizontal component) and **northing** (vertical component). Locations are uniquely described by **UTM zone number**, **easting** (distance from the central meridian of the zone, meters) and **nording** (distance from the equator, meters).



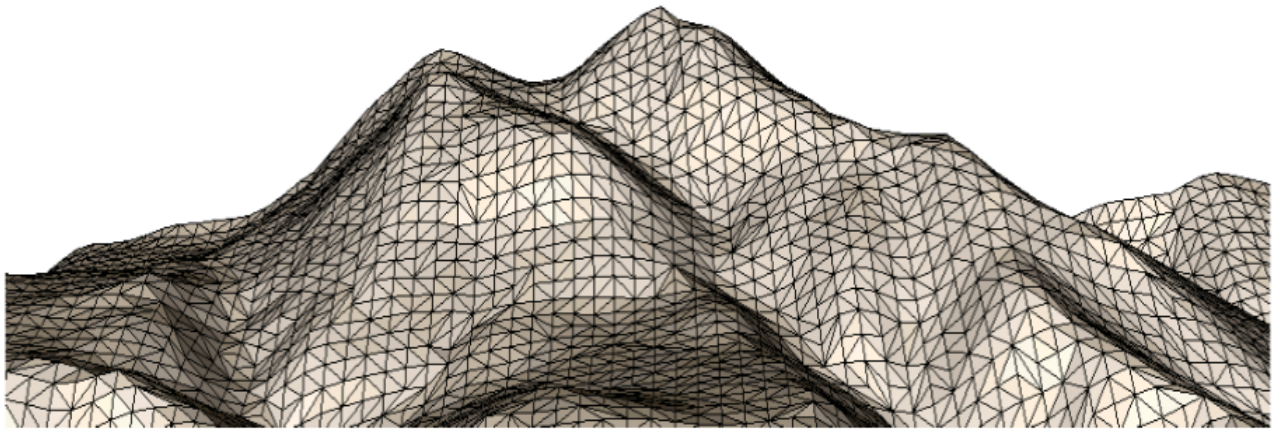
Mercator Map Projection.



UTM zones for the African Continent.

Geo-spatial Data Formats

- **Line (polyline) data**: linear paths that connect multiple points. Used to model things like river networks. Each line segment in a polyline is defined by at least two geographic coordinates.
- **Polygon data**: areas or boundaries, connected multiple points to form a **closed shape**. The first and last points must be the same.
- **Triangular Irregular Network (TIN)**: vector-based representation used to model surfaces. Consists of a series of non-overlapping triangles that are formed by connecting points with known elevation values.



Representation of topography using Triangular Irregular Networks (TINs)

- **Vector data**: lines, polygons, TINs are vector data - they represent geographic features as discrete geometric shapes.
  - **Time-stamped raster data**: continuous variables, e.g., temperature, that change over time.
  - **Time-stamped vector data**: time-stamping features like points lines or polygons. E.g., changing boundaries of a floodplain over a course of a storm.
- **Raster data**: **grid-based** representation of continuous spatial phenomena. E.g., each cell in the grid can cover 1km^2 area.
- **Trajectory data**: **movement of objects over time**, capturing both spatial and temporal dimensions. Each trajectory is composed of a sequence of time-stamped geographic points as the object moves through space over time.

File Types in Geospatial Data

1. **Shapefile (.shp)**: stores basic vector data (points, polylines, polygons) along with attribute information. Not human-readable (binary). **Do not support TINs or raster data**. Used for **static data** or **static time-stamped snapshots**.

Requires **.shx**: index file to link geometry and attribute data; and **.dbf**: attribute data file containing tabular information related to the features.

- **GeoJSON (.geojson)**: stores basic **vector data** (points, polylines, polygons) and supports **attribute information**, **time-stamped**, and dynamic data. Human-readable (JSON format) and **lightweight**, ideal for web applications. Uses WGS84 coordinate system. Does not support raster or TIN data. Self-contained and widely used for tracking changes or movements.
- **KML (.kml)**: XML-based format for geospatial data, including **vector data** and 3D geometries. Supports **styling** (colors, icons) and time-stamped/dynamic data. Ideal for **web applications** and virtual globes (e.g., Google Earth). Does not store raster or TIN data directly but **can reference external images** or **overlays**.
- **GeoTIFF (.tif/.tiff)**: **raster data** with **geo-referencing metadata** (CRS, resolution, origin). Suitable for satellite imagery, DEMs, and grid-based data. Supports **continuous** (e.g., elevation) and **categorical data** (e.g., land cover). Does not support vector or TIN data. Allows **multiple data bands in a single file** (e.g., RGB, infrared).
- **OBJ (.obj)**: stores **3D geometric data** (meshes, surfaces, TINs) as plain text, widely supported in 3D modeling. No georeferencing or time/attribute metadata. Focuses on 3D geometry. Commonly used for 3D terrain models and TINs.

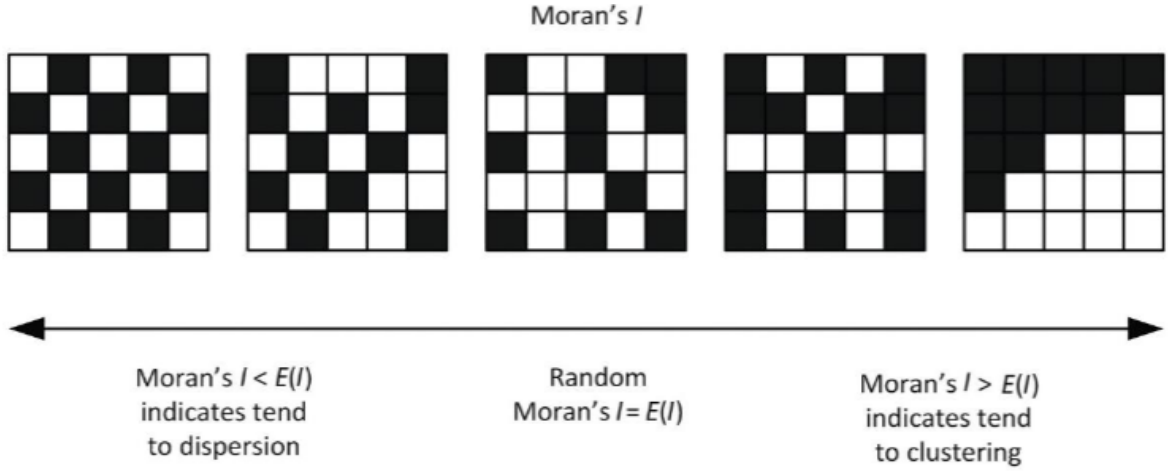
Spatial Statistics

Spatial statistics is a branch of statistics dedicated to analyzing data tied to spatial locations. It incorporates **spatial dependence**, i.e. spatially proximate data points are often more alike than distant ones. **Geostatistics** is a subfiled of spatial statistics focusing on modeling and prediction of **spatially continious phenomena** (temperature fields).

Spatial Autocorrelation

Correlation of a variable **with iteslf through space**. There is **positive** (geographically proximate locations have similar values), **negative** and **no spatial autocorrelation**. Spatial autocorrelation can be measured by:

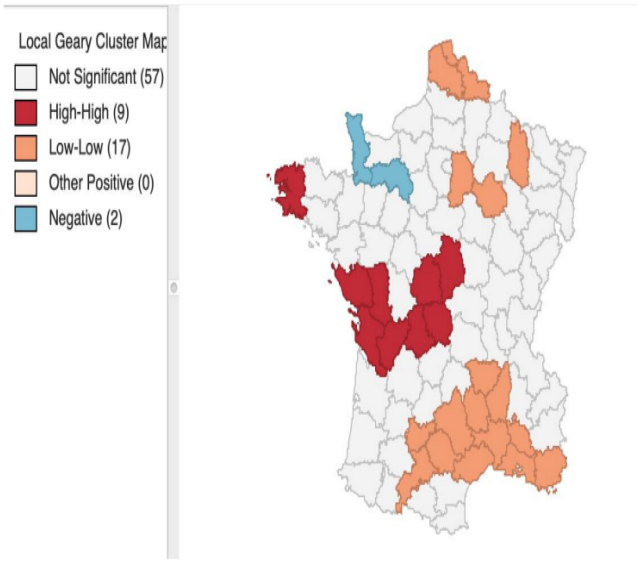
1. **Global measures**: assess the overall pattern of spatial dependence across an entire area. Provide **singular** summary value that indicates whether spatial data exhibit **clustering**, **dispersion** or **randomness**. Fail to capture local variations, clusters, outliers.
   - **Moran's I**: evaluates the overall similarity between values at neighboring locations:
   - $I = \frac{N}{W} \times \frac{\sum_i \sum_j w_{ij}(x_i - \overline{x})(x_j - \overline{x})}{\sum_i (x_i - \overline{x})^2}$
   - N is the number of observations, x_i and x_j are the values at locations i and j, $\overline{x}$ is the mean of the variable, w_{ij} is the spatial weight (strength of the spatial relationship betwen locations i and j), W is the sum of weights.
   - Ranges from -1 (perfect dispersion) to $+1$ (perfect clustering).
   - w_{ij} can be defined based on distance (usually **inverse distance** giving more weight to closer points) or other criteria such as adjacency.



Moran's I as a global measure of spatial autocorrelation.

1. **Local measures (local indicators of spatial association (LISA))**: assess spatial autocorrelation at a specific location or small region within the study area. Local measures help identify **clusters** (hot spots, cold spots) and **spatial outliers**.
   - **Local Moran's I**: local version of Moran's I.
   - $I_i = \frac{(x_i - \overline{x}) \sum_j w_{ij}(x_j - \overline{x})}{\sum_i (x_i - \overline{x})^2 / N}$

- $I_i > 0$: similar values cluster together.
- $I_i < 0$: dissimilar values cluster together.
- **High-high clusters**: high value location is surrounded by neighbors with high values (**hotspot**).
- **Low-low clusters**: low value location surrounded by other low values (**cold spot**).
- **High-low** or **low-high** clusters: spatial outliers. Indicate significant local deviations from surrounding spatial context.
- **Statistically insignificant**: Local Moran's I value doesn't show a strong or reliable spatial autocorrelation (no clear pattern of clustering).



Sample map showing local spatial correlation.

Point Pattern Analysis

Point pattern analysis in geostatistics is the study of the **spatial arrangement or distribution of points**. Such points can represent the **existence of an object or event** at a specific location. Point pattern analysis seeks to determine whether such objects or events are **randomly distributed**, **clustered**, or **regularly spaced (dispersed)** within the study area.

- **Quadrat analysis**: determine whether a spatial distribution of points is random, clustered or dispersed.
  1. Divide into smaller equally sized regions, **quadrats**.
  2. Count the number of points in each quadrat and get the distribution of points.
  3. You can count **mean** number of points per quadrat i, **variance**, **variance-to-mean ratio (VMR)**.
     - VMR ≈ 1: points are randomly distributed.
     - VMR > 1: points are clustered (more variability in the quadrat counts than expected under randomness).
     - VMR < 1: points are dispersed.
  4. **Chi-square** test for randomness:
     1. $\mathcal{X}^2 = \sum_{i=1}^{N} \frac{(x_i - \overline{x})^2}{\overline{x}}$
     2. Compare $\mathcal{X}$ with critical value from a chi-square distributuion table with $N-1$ degrees of freedom to access whether the observed distribution is different from a random distribution.
  5. **Hypothesis testing**:
     1. H_0: the point pattern follows a random (Poisson) distribution.
     2. H_1: does not follow random distribution.
     3. If calculated chi-square value exceeds the critical value, the null hypothesis is rejected.

Spatial Interpolation

Spatial interpolation is a technique used to **estimate values at unsampled locations** based on the values of **nearby sampled points**. Is assumes that **points closer together** are more likely to have **similar values** (**spatial autocorrelation**). Interpolation allows for the estimation of values across the entire study region, filling in gaps between sampled points.

Kriging

Kriging estimates unknown values at unsampled location and provides a measure of the uncertainty associated with those estimates.

- $Z^*(s_0) = \sum_{i=1}^{n} \lambda_i Z(s_i)$ - linear combination of the known values at nearby locations with $Z^*(s_0)$ - value at an unsampled location s_0.
- weights λ_i are chosen based on the spatial structure of the data represented by **semivariogram**.

Semivariogram measures how the similarity between points changes as the distance between them increases, reflects how the variance between point values changes with increasing separation distance.

- $\gamma(h) = \frac{1}{2}\mathbb{E}\left[(Z(s_i) - Z(s_i + h))^2\right]$ with h distance between two points.
- $\hat{\gamma}(h) = \frac{1}{2N(h)} \sum_{i=1}^{N(h)} \left[(Z(s_i) - Z(s_i + h))^2\right]$ - **empirical semivariogram** (because $\mathbb{E}$ is unknown) with $N(h)$ number of data point pairs separated by distance h.

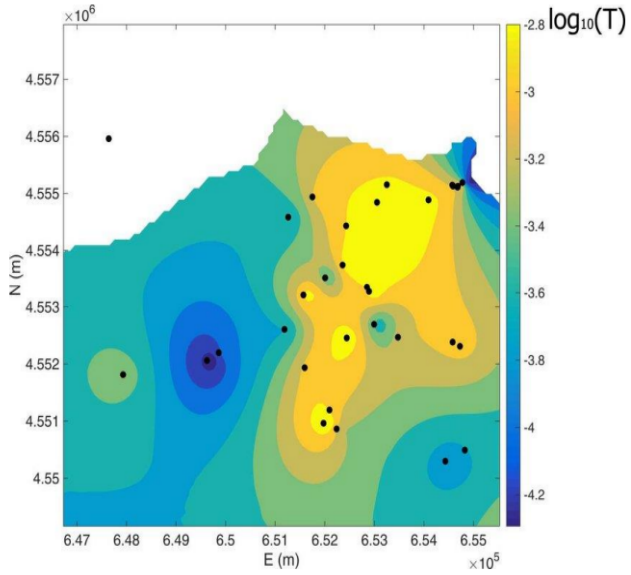
Ordinary kriging equations derived by minimizing the kriging varinace (variance of prediction error) while ensuring that the sum of the weights equals to 1. Used to calculate weights λ_i:

- $\sum_{j=1}^{n} \lambda_j \gamma(s_i - s_j) + \mu = \gamma(s_i - s_0)$ for i=1,...,n with s_0 unknown location and μ Lagrange multiplier used to enforce the unbiased constraint.
- $\sum_{j=1}^{n} \lambda_j = 1$

Algorithm to solve ordinary kriging equations for weights λ:

1. **Compute semivariogram values**: use $\hat{\gamma}(h)$ to compute semivariogram values for all pairs of known points $s_1, \ldots, s_n$ as well as semivariogram values between each known point and the prediction location s_0.
2. **Set up the kriging matrix**:
    1. Construct a matrix of semivariogram values between all known points. The matrix is symmetric with values $\gamma(s_i - s_j)$.
    2. Create a vector of semivariogram values between each known point and the prediction location s_0 denoted as $\gamma(s_i - s_0)$.
3. **Solve the kriging equations**: the matrix system is solved using linear algebra techniques, like matrix inversion.
4. Weights λ_i determine how much influence each known point s_i has on the prediction at s_0. Points closer to s_0 generally have higher weights.

$$\begin{bmatrix} \gamma(s_1 - s_1) & \ldots & \gamma(s_1 - s_n) & 1 \\ \gamma(s_2 - s_1) & \ldots & \gamma(s_2 - s_n) & 1 \\ \vdots & \ddots & \vdots & 1 \\ \gamma(s_n - s_1) & \ldots & \gamma(s_n - s_n) & 1 \\ 1 & \ldots & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} \lambda_1 \\ \vdots \\ \lambda_n \\ \mu \end{bmatrix} = \begin{bmatrix} \gamma(s_1 - s_0) \\ \ldots \\ \gamma(s_n - s_0) \\ 1 \end{bmatrix}$$



Example output of Ordinary Kriging interpolation.

Kriging Variants

1. **Ordinary kriging** (discussed above):
    1. Assumes constant mean of data.
    2. Weights are determined based solely on spatial autocorrelation modeled by semivariogram.
    3. Most used form of kriging.
2. **Universal kriging**:
    1. Accounts for spatial trend in the data, i.e. mean is not constant.
    2. Incorporates deterministic trend and spatial autocorrelation in the model.
    3. Used when there is a clear trend in the data (e.g., elevation increasing with latitude).
3. **Cokriging**:
    1. Multivariate extension of kriging that interpolates multiple correlation variables simultaneously.
    2. Uses correlation between variables to improve the estimation of the target variable.

Spatial Regression

Spatial regression is used to model relationships between a dependent variable and one or more independent variables while explicitly accounting for the spatial arrangement of data points (traditional regression models assume that observations are independent of each other, but we have spatial autocorrelation).

Geographically Weighted Regression (GWR)

GWR allows for the estimation of local (rather than global) relationships between variables, unlike traditional regression models, which assume that the relationship between independent variables (predictors) and the dependent variable is the same across all locations.

1. **Local parameter estimation**: GWR estimates a separate set of regression coefficients for each location in the study area. The idea is that the relationship between the dependent and independent variables may change across space.
2. **Spatial weights**: uses spatial weights to assign **more influence** to **nearby observations** when estimating the local regression coefficients.

$$y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_k x_{ik} + \epsilon_i \text{ - general form}$$

$$y_i = \beta_0(u_i, v_i) + \beta_1(u_i, v_i) x_{i1} + \beta_2(u_i, v_i) x_{i2} + \cdots + \beta_k(u_i, v_i) x_{ik} + \epsilon_i \text{ - GWR form}$$

- (u_i, v_i) coordinates (e.g., latitude and longitude) of location i.
- $\beta_k(u_i, v_i)$ regression coefficinets that vary with location.
- ϵ_i error term at location i.
- $x_{i1}, \ldots, x_{ik}$ independent variables (predictors).

Each location i has its own **local regression**, where the weights are assigned to the observations in the dataset depend on their proximity to location i. **Weighting function** w_{ij} can be any distance-based function, most common choice is **Gaussian kernel**:

$$w_{ij} = \exp\left(-\frac{d_{ij}^2}{2h^2}\right)$$

- d_{ij} distance between location i and j.
- h bandwidth, width of the neighborhood (how much weight is given to distant points).

For each location (u_i, v_i) GWR solves a weighted least squares regression problem, where the weights are derived from the spatial proximity of other data points:

$$\hat{\beta}(u_i, v_i) = (X^T W_i X)^{-1} X^T W_i y$$

- X matrix of independent variables.
- y vector of observed values for the dependent variable.
- W_i diagonal matrix of spatial weights for location i, where each element corresponds to weight w_{ij} for observation j.

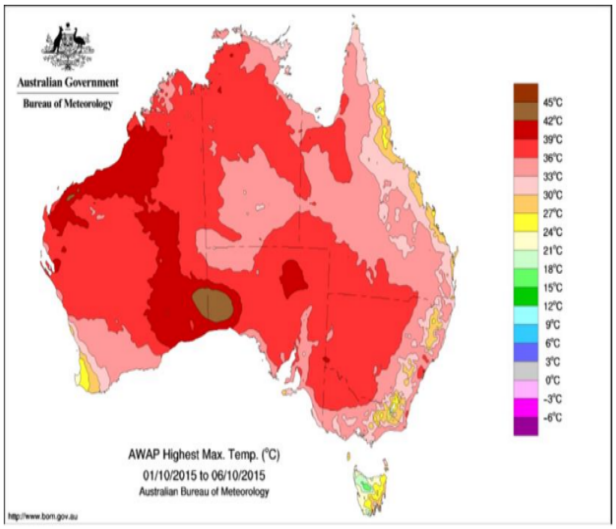
Python-based Geostatistics

1. **PySAL (Python spatial analysis library)**: supports **spatial autocorrelation** (local and global, .e.g., Moran's I and LISA), **spatial regression** (e.g., GWR), **point pattern analysis**, **spatial interpolation** (e.g., kriging and inverse distance weighting).
2. **GSTools**: focuses on kriging and variogram modeling.
3. **Geopandas**: integrates well with PySAL.

Geospatial Data Visualization

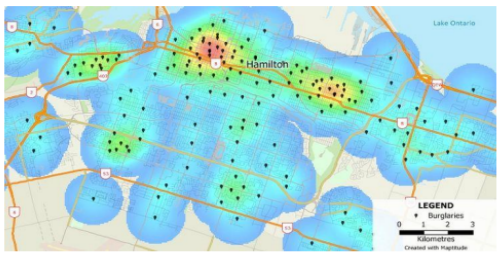
Geospatial visualization includes specialized methods like heat maps, choropleth maps, and 3D terrain models, and requires handling geographic projections, spatial relationships, and layers of location-based data.

- **Overlaying**: placing spatial data (e.g., points, lines, or polygons representing things like locations, roads, or regions) on top of a base map that shows real-world features such as streets, buildings, terrain, or satellite imagery. Both maps should use same geographic coordinate system or projection.
- **Choropleth maps**: represent the distribution of a variable across predefined regions, such as countries, states. Geographic areas are filled with varying shades or colors based on the value of the data associated with that area (darker colors mean more intense values). Usually used for **discrete** variables.
  1. **Equal interval classification**: divides range of data into equal-sized intervals. Suitable when range is uniform.
  2. **Quantile classification**: divides the data so that each class contains equal number of data points (e.g., 100 points, 5 classes $\rightarrow$ 20 points for each class).
  3. **Standard deviation classification**: divides data based on how much values deviate from the mean. Useful for visualizing outliers and mean of data.

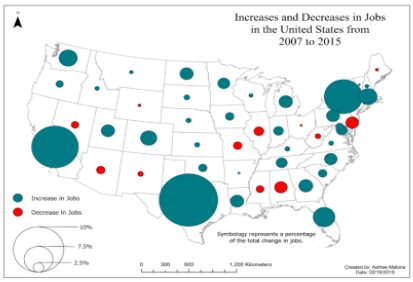


An example of a Choropleth Map.

- **Contour maps**: represent continuous data over a 2D plane, where lines (contours) connect points of equal value. Used to visualize data such as elevation, temperature or pressure.
- **Hotspot maps**: focus on showing where events occur frequently. Used to identify **clusters**.
- **Proportional symbol maps**: use symbols (e.g., circles, squares) whose size varies in proportion to the value of the data being represented. Larger symbols indicate higher values.



An example of a hotspot map showing incidents of burglary in a city.



An example of Proportional Symbol Maps showing increase or decrease in jobs in the US.

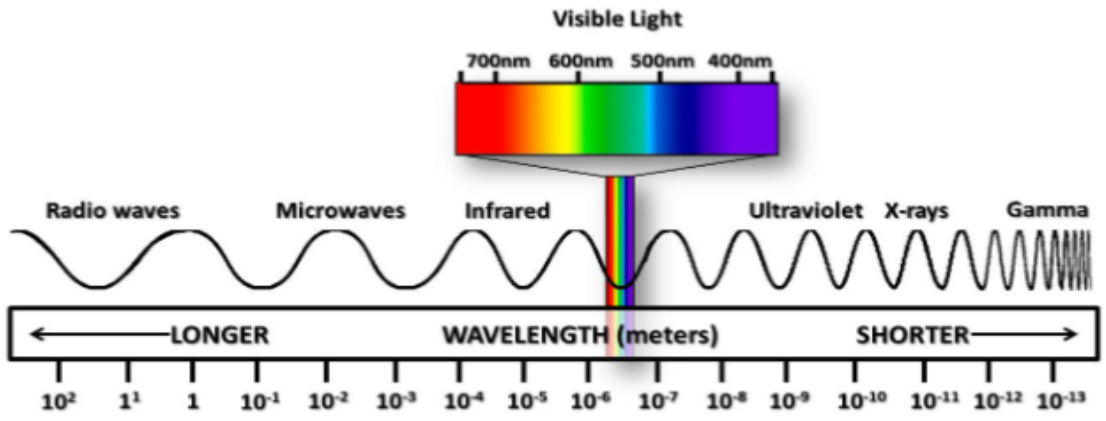
7: Remove Sensing (RS)

Remote sensing (RS) data refers to the acquisition of information about the Earth's surface and atmosphere from a distance, typically using platforms such as satellite, aircrafts, drones, or Balloons. RS data can be represented as **raster** geospatial data, **point clouds**, and **vector data**, representing geographic features as points, lines, polygons. RS is important:

1. **Access to challenging areas**: provides data where in-situ observarions are difficult to get (dense forests).
2. **Global reach**: satellites orbit the Earth.
3. **Economies of scale**: once satellite is lauchnes, it covers vast areas at a fraction of the cost compared to extensive ground-based campaigns.
4. **Repeatability and consistency**: satellites frequently revisit over the same area.
5. **Long-term records**: archives of satellite data extend back several decades.
6. **On-demand data from aerial platforms**: drones and aircraft can be deployed as needed.
7. **Non-intrusive data collection**:
    1. **Minimal environmental impact**: RS data collection doesn't distrub the environment.
    2. **Safety and convenience**: RS methods eliminate the risks associated with fieldwork.
8. **Vertical profiling with balloons**: balloons provide vertical profiles of the atmosphere.

RS Terminology

- **Electromagnetic radiation**: energy emmited and propagated through space in the form of waves. It is the medium through which sensors detect information. Radiation can be reflected, absorbed or emitted by objects, and sensors can capture these interactions. The spectrum includes, in order of increasing wavelength and decreasing frequency, radio waves, microwaves, infrared radiation, visible light, ultraviolet radiation, X-rays, gamma rays.
    - **Wavelength**: distance between consecutive peaks of a wave, measured in meters, nanometers. Shorter waves (like visible light) capture fine details of the Earth's surface, longer wavelength (infrared) penetrate clouds and provide information on thermal properties.
    - **Frequency**: number of times a wave repeasts in one second, measured in hertz (Hz). Higher frequency means more energy and typically shorter wavelengths, shorter frequency means the opposite.
- **Electromagnetic spectrum**: complete range of electromagnetic radiation, encompassing all wavelengths and frequences. Each type of radiation interacts with matter in distinct ways.

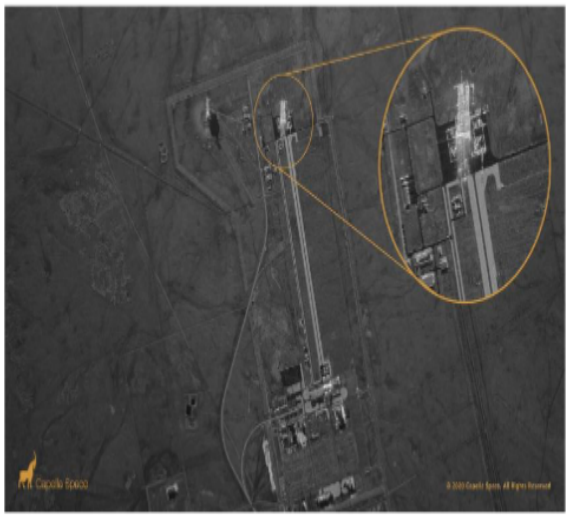


The Electromagnetic Spectrum.

- **Wavelength bands**: ranges of wavelengths within the electromagnetic spectrum that sensors can detect. E.g., visible band (400-700 nm) used to capture images similar to what human eye can see; near-infrared band (700-1400 nm) used to monitor vegetation health.
- **Single-band (-channel) imaging**: capture data in only 1 specific band. E.g., **thermal infrared imaging**, 3-14 micrometers, detect heat and temperature variations.

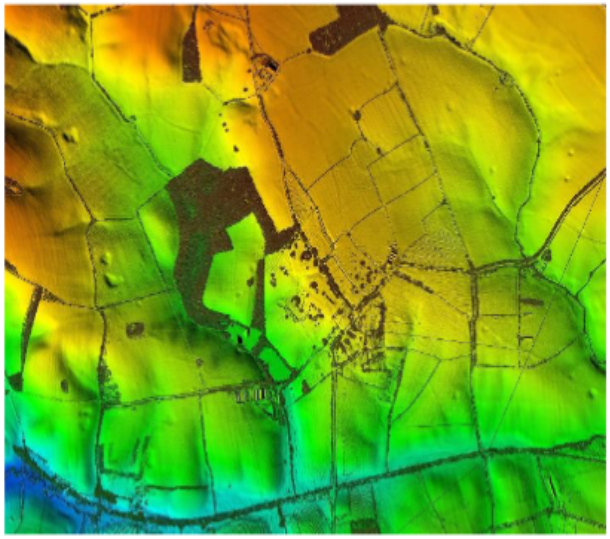
The right image shows the same area as the left image captured with an aerial thermal infrared sensor.

- **Microwave and radar sensing**: involve the use of microwaves (longer) to penetrate clouds, vegetation, ground. **Synthetic aperture radar (SAR)** microwave technology provides topographic maps.



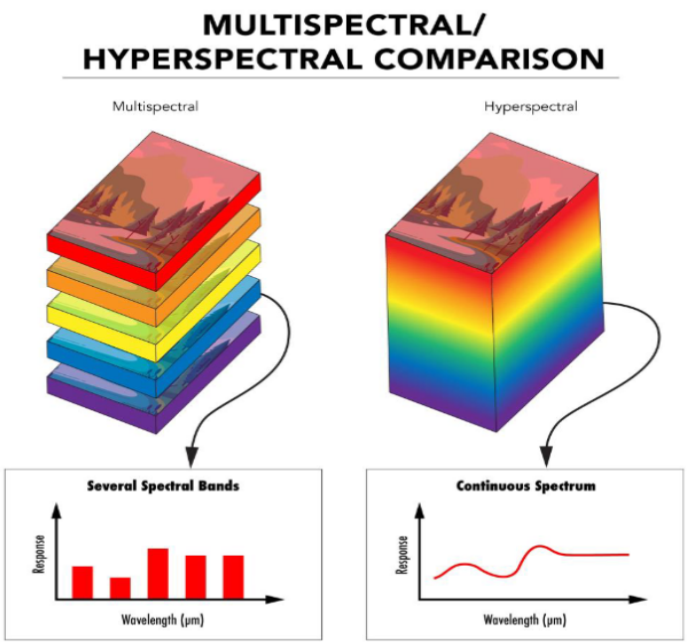
Example of a Synthetic Aperture Radar (SAR) image.

- **LiDAR (light detection and ranging)**: uses laser pulses to measure distances and generate high-resolution, three-dimensional maps of surface. Operates in near-infrared region, 800-1550 nm.



Example of a LiDAR image.

- **Multispectral imaging**: capture data from multiple wavelength bands. (e.g., green, blue and near-infrared). Provides balance between detail and coverage.
- **Hyperspectral imaging**: capture from hundreds of narrow, contiguous wavelength bands. Useful in applications requiring precise material discrimination, like agricultural assessments.



Conceptual illustration of Hyper-spectral (left) vs multi-spectral (right) images.

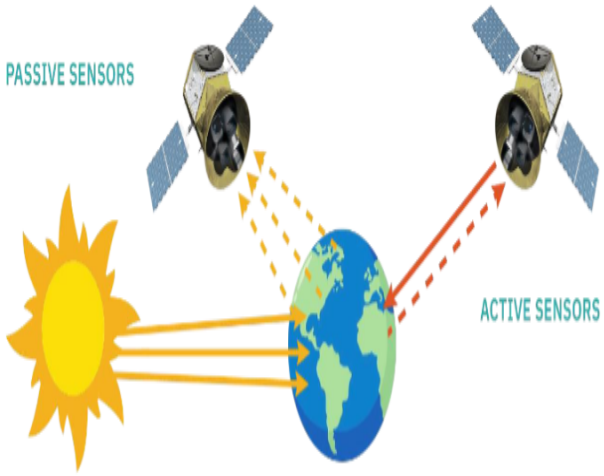
Types of Platforms

Platform	Spatial Resolution	Temporal Resolution	Coverage
Satellite	High to low*	Consistent data over extended period, ranges from daily to monthly; some satellites provide multiple observations per day	In many instances global or continental
Aerial	High to very high*	On-demand, but sometimes consistent, in the latter case frequency depends on flight schedules.	Variable, often in the order of 100 to 1000s of km^2
Drone	Very high*	On-demand, but sometimes consistent	Variable, often in the order of 10s to 100s of km^2
Balloon	High*	Variable, can provide long-duration observations of the same area	Variable, often in the order of 10s to 100s of km^2

- Low spatial resolution: $>$ 1 kilometer (1000 meters) per pixel.
- Medium spatial resolution: 10 to 1000 meters per pixel.
- High spatial resolution: 1 to 10 meters per pixel.
- Very high spatial resolution: $<$ 1 meter per pixel.

Types of Sensors

- **Passive sensors**: detect natural energy that is emitted or reflected by the observed objects. The most common example of a passive sensor is a camera, which captures visible light reflected from the Earth's surface.
- **Active sensors**: emit their own energy and measure the amount of that energy reflected back from the target (LiDAR, RADAR (radio detection and ranging)).



Passive vs. active sensors.

Preprocessing Remove Sensing Data

- **Georeferencing**: process of assigning geographic coordinates to raw remote sensing images to align them with a known spatial reference system, which is done using **ground control points (GCPs)**. The selected points should be easily identifiable in remove sensing imagery. Then match the positions of GCPs in the raw imagery to their real-world coordinates, allowing each coordinate to be assigned to each pixel in the RS images.
- **Cloud masking**: identify and remove or flag cloudy areas.
- **Atmospheric correction**: corrects systematic distortions caused by atmospheric particles, like **ozone**, which can absorb the light.
- **Radiometric correction**: adjust the brightness and intensity values in the imagery to correct for sensor noise. Ensures that the pixel values accurately represent the reflectance of the Earth's surface.
- **Geometric correction**: correct distortions caused by the sensor's perspective, Earth's curvature, topographic variations.
- **Mosaicking**: combine multiple RS images into a composite image.
- **Calibration**: ensure that raw data from sensors is consistent and comparable over time and across different sensors.

RS Data Levels

- **Level 0: raw data**: unprocessed data directly from sensors, often in binary format.
- **Level 1: georeferenced data**: data is callibrated radiometrically and geometrically, corrected for sensor-specific distortions and is georeferenced.
- **Level 2: derived products**: derived specific geophysical variables from level 1 data and data is corrected for atmospheric effects (e.g. sea surface temperature).
- **Level 3: mapped products**: mapped onto a uniform grid. Involve spatial and/or temporal aggregation. E.g., global monthly averages of sea surface temperature.
- **Level 4: model outputs and analysis**: model generated outputs.

8: Point Clouds

- Common types of 3D data in EDA are: **point clouds**, **3D meshes**, **TINs**, **digital elevation models (DEMs)**, **3D volumetric data (voxels)**, **cross-sections** and **profiles**.
- **Point cloud data** is a raw unstructured dense collection of points in 3D space that represent the surface geometry of objects and environments, which is very adaptable for ML and statistical algorithms. Each point has spatial (x, y, z) coordinates, which may also include additional features, like color, classification labels, etc.
- LiDARs generate point clouds as their primary output, and point clouds contain a lot of detail and serve as the basis for generating other 3D data types, such as meshes.

Characteristics of Point Clouds

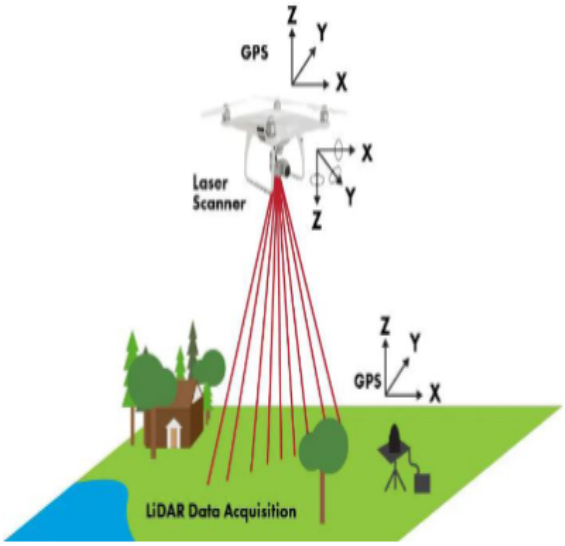
- **Unstructured**: no connection between points (unlike grids or meshes).

- **Unordered**: no sequence to the points, order can be changed without changing the cloud.
- **No** assumptions about the **geometry**.
- **Huge dataset** when capturing **fine-grained** surface..
- **3D + additional features** as dimensions.
- **Sparse** and **unevenly distributed**, with some regions densely sampled.
- Contain **noise** and **outliers** due to sensor inaccuracies, environmental conditions, reflections, occlusions.
- **Dynamic point clouds**, e.g., moving objects or time-series scans.

Point Cloud Generation

LiDAR (Light Detection and Ranging)

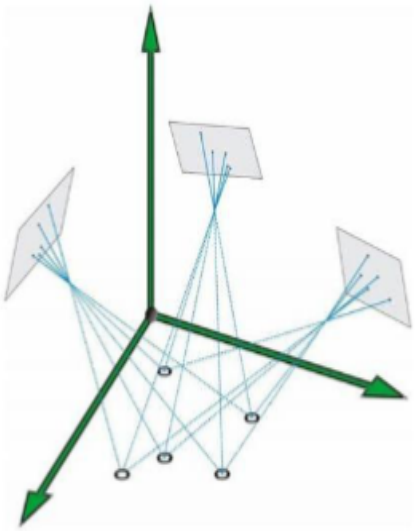
- A laser scanner emits pulses of light and records the time taken to return after hitting a surface (="time of flight").
- Using time of flight and speed of light LiDAR calculates distance to the surface.
- LiDAR then determines the 3D position of each point based on the calculated distance and the angle at which the laser pulse was emitted (using internal sensors to track the LiDAR system orientation and position).
- This results in a spherical coordinate system which is then used to compute x, y, z coordinates of the point in a 3D Carterisan coordinate system.



LiDAR scanning.

Photogrammetry

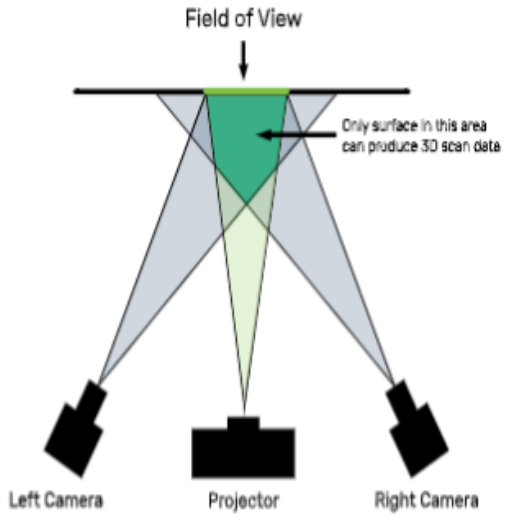
- Photogrammetry computes 3D points indirectly by analyzing and triangulating cooresponding features from multiple overlapping images.
- It relies on identifying and matching points in 2D **overlapping images** of the **object** or scene from **different angles**.
- Then it uses e.g. perspective projection and camera pose estimation to reconstruct object 3D coordinates.



Schematic of how Photogrammetry works.

3D Structured Light Scanners

- 3D structured light scanners create point clouds by **projecting** a known pattern of **light**, such as stripes or grids, **onto an object's surface** and capturing the deformed pattern with one or more cameras.
- The **deformation** is analyzed to determine how the **object's surface alters the light**, and **triangulation** is used to **calculate the 3D coordinates** of points on the surface.
- Structured light scanners are highly precise and fast.



3D structured light scanning

Synthetic Sources

- Point clouds can also be simulated using computational models or derived from meshes or voxels.

Platforms for Point Cloud-based Observations

1. **Satellite-based point clouds**: satellite misions equipped with LiDAR or photogrammetry can produce large-scale point cloud data. Use cases include **regional ecosystem monitoring** (forest biomass, canopy height, carbon storage on a continental scale), **topographic mapping**, **cryosphere studies**.
2. **Airborne point clouds (planes)**: airborne LiDAR systems mounted on planes, more flexible than satellite. Use cases include **urban planning**, **flood risk mapping**, **forest management**.

3. **Drone-based point clouds**: drones equipped with LiDAR or photogrammetry cameras can capture extremely high-resolution point clouds. Use cases are **agriculture**, **coastal monitoring**, **archaeological studies**.
4. **Ground-based point clouds**: handheld and vehicle-mounted LiDAR scanners for localized environments. Use cases are **infrastructure assessment**, **urban street planning**, **ecological surveys**.

Preprocessing of Point Clouds

- **Denoising**: noise is small, random inaccuracies or distortions coming from e.g. sensor imperfections, environmental conditions, object movement. Noise doesn't distort the structure, only slightly the surface. Denoising is the process of adjusting noisy coordinates or features to align more closely with their true values.
- **Removing outliers and invalid points**: outliers are points or clusters of points that deviate significantly from general data distribution. They appear due to sensor errors, occlusions, temporary obstructions. Methods to remove outliers are:
  - **Statistical outlier removal**: based on distance thresholds or neighborhood density.
  - **Radius-based filtering**: eliminating points with insufficient neighbors within a radius.
  - **Clustering algorithms**: detecting points that do not belong to major clusters.
- **Alignment and registration of point clouds**: it involves combining multiple points clouds into a single, inified coordinate system, like combining multiple scans to create a complete and continious 3D model. **Starts** with **aligning** the point clouds using methods that match **recognizable features**. Then, **fine-tune** the alignment using **iterative closest point (ICP)**, adjusting point clouds to minimize gaps and mismatches (e.g. **shifting**, **rotating** the clouds).
- **Downsampling**: reduce the number of data points while preserving overall structure and details. **Voxel grid downsampling**, **uniform sampling**, **random sampling**.
- **Normalizaing and scaling**: fit within a coordinate system or range, e.g., shifting centroid to origin $(0, 0, 0)$ and scaling to fit within a unit sphere.
- **Feature enginnering**:
  - **Local features**: immediate geometry and spatial relationships around a point. E.g., point density, number of neighboring points within a specified radious of each point.
  - **Aggregated features**: summarize information across larger regions or entire point cloud. E.g., mean, variance, bouding box.
  - **Attribute-based features**: derived from additional data, such as color, classifications. E.g., color histograms or dominant color clusters to summarize the distribution of colors in a region.

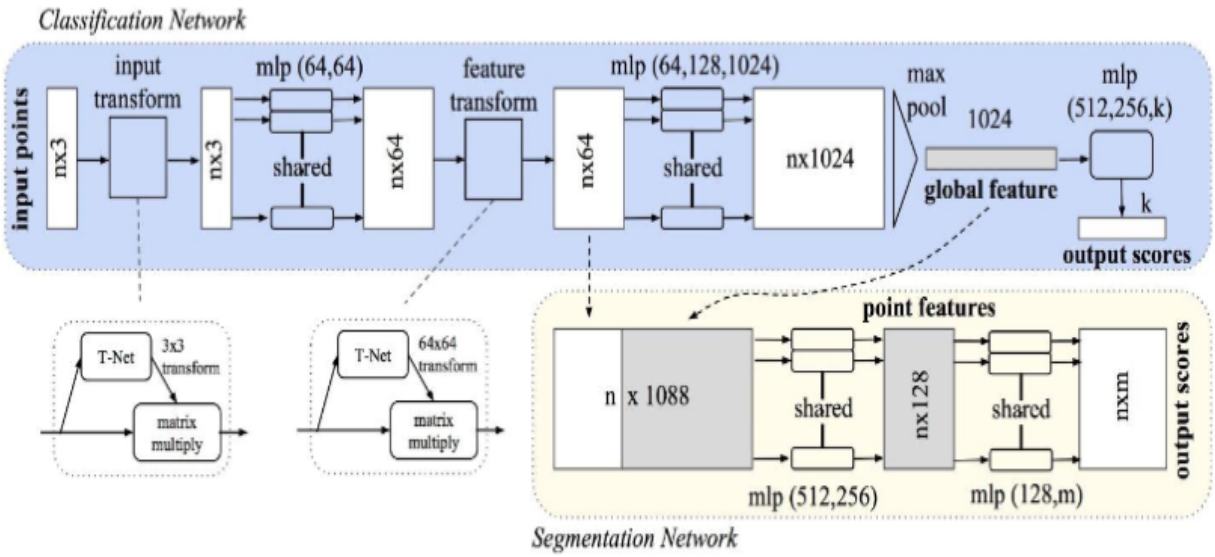
Fusing Point Clouds with Other Data Modalities

1. Align both datasets to the same geographic coordinate system, using GPS data, ground control points, sensor metadata or georeferencing.
2. Use spatial location of each point in the point cloud data to identify its corresponding instance in another dataset, like image pixel.
3. Extract additional data and assign it to the respective points in the cloud data.

ML on Point Cloud

- Since point cloud data is unordered, ML models should exhibit **permutation invariance**, i.e. the output of the model should not depend on the order in which the points are presented in the input data. Thus, CNNs is not a good model choice.
- **PointNet** is a model developed specifically for point cloud data. PointNet **aggregates features** from **all points** using symmetric operation, e.g., **max-pooling** layer, ensuring that the **order of the points does not affect the final global feature representation**.
  1. Represent each point as a vector of coordinates (x, y, z) with additional features like color (r, g, b). Get $N \times d$ vectors (N points, d dimensionality of each point).
  2. Apply **shared MLPs** independently to each point. **Shared MLPs** use the same weights for different parts of the input or across multiple inputs. This significantly reduces the number of parameters in the model, which can help prevent overfitting and make the model more memory-efficient. Get features matrix $N \times K$, N K-high-dimensional feature vectors. **Weight sharing ensures permutation invariance**.
  3. **Aggregate** $N \times K$ feature matrix into a global feature vector using e.g. **max pooling**. Get K-dimensional global feature vector **summarizing the entire point cloud**.
  4. Depending on the task, process this K-dimensional vector. E.g., for classification pass it through a FC layer to predict class label for the entire point cloud. For **segmentaiton**, both global and individual point features are combined via concatenation to predict class label for each point.
- **Dynamic Graph CNNs (DGCNN)** is another model built for point clouds. It uses a graph over the points based on their spatial relationships and apply **graph convolutions**, which operate on the graph structure and are **invariant to**

the permutation of points.



The PointNet architecture for Point Clouds.

9: Geospatial ML

Geospatial ML refers to the application of ML techniques to data that contains geographic components, e.g., latitude and longitude, satellite imagery, point clouds, raster data, etc.

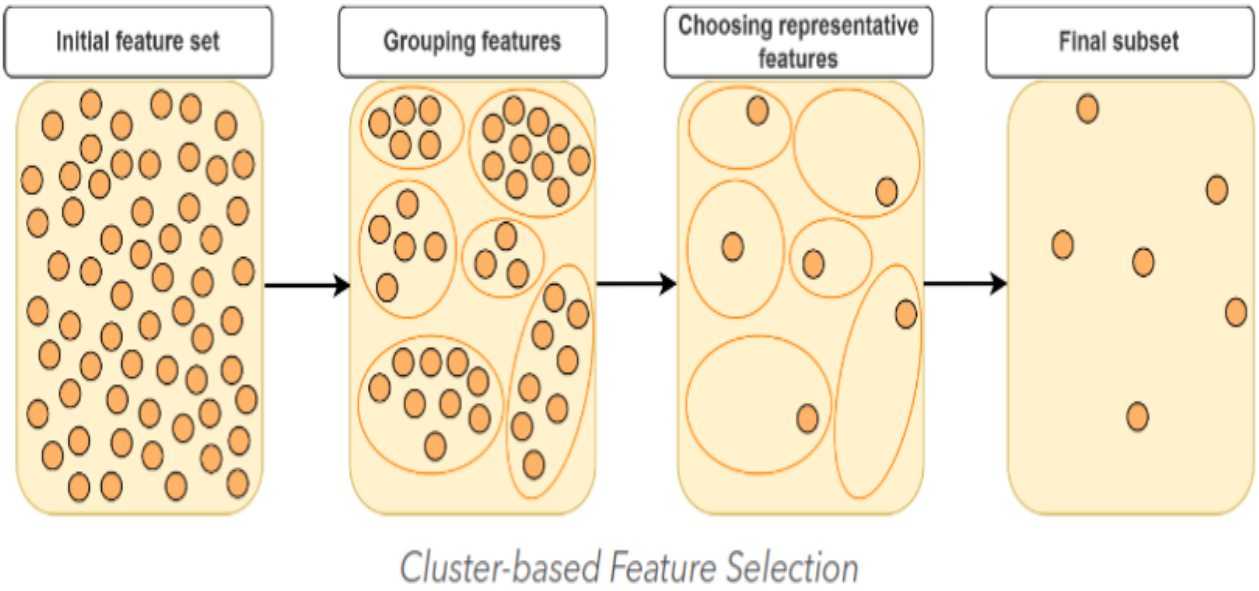
Feature Engineering in Geospatial ML

- **Feature engineering** is the process of selecting, modifying, or creating new features from raw data to improve a ML model's performance. It helps identify **most relevant features, remove irrelevant and redundant features, reduce dimensionality**, making training more efficient and model less prone to overfitting.
- **Feature learning** is the process where DL model discovers and refines the most useful features on its own: we pass raw data to the model and it learns by itself which data is the most relevant for the prediction.
- **Spatial feature engineering** is a subset of feature engineering focused on creating features from data with spatial or geographical components. It leverages spatial - such as proximity, neighborhood effects, spatial autocorrelation - while using location specific tools and techniques to extract meaningful insights for location-dependent applications.
- **Map matching** refers to connecting different geographical datasets, while **map synthesis** refers to the use of geographical structure to derive new features from a given dataset.

Feature Selection

Feature selection is the process of **choosing the most relevant inputs** (features) for a model while discarding those that may be redundant, irrelevant, or noisy, helping the model focus on high-quality data for better performance.

1. **Cluster-based feature selection**: measure similarity between features (correlation, mutual informaiton), cluster features using e.g. k-means, select representative feature (most central or with highest variance) in each cluster.
2. **Recursive feature elimination**: train a model, assess the importance or contribution of each feature, remove the lest important features, retrain. Repeat recursively until a specified number of features remain or model perfomance no longer improves. In DNNs, use **gradient-based analysis** to compute importance scores. By computing the gradient of the output w.r.t. each input feature, assess the sensitivity of the models' predictions to changes in each feature. High gradients and small changes in a feature → strong effect on the output → high importance.

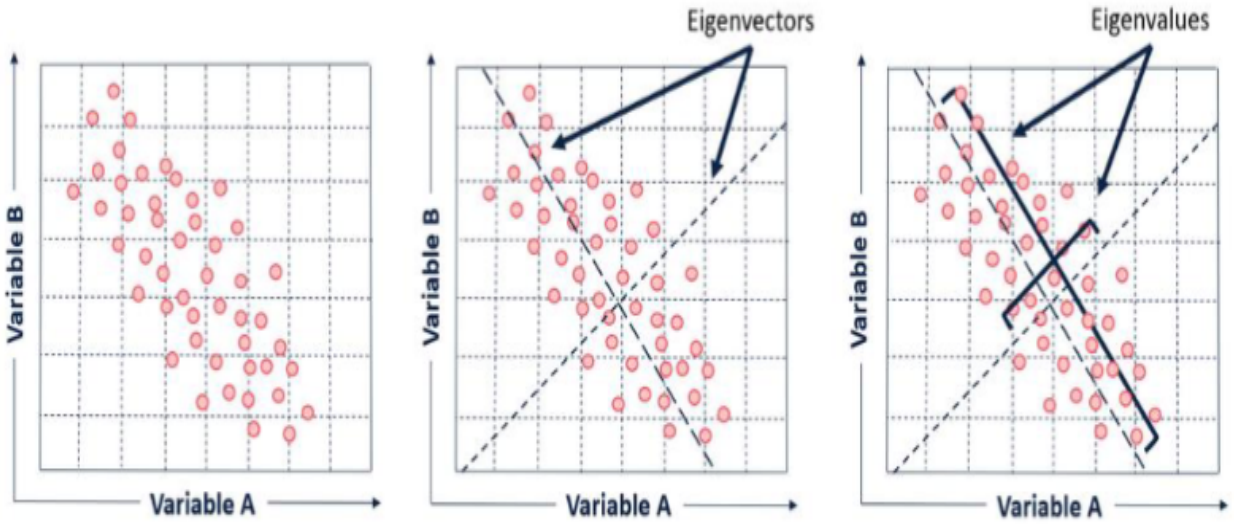


Feature Combination

- Feature combination is the process of creating new features by combining existing ones to capture more complex relationships in the data.
- **Multiplying or dividing features**: e.g., diving total annual rainfall by the number of rainy days → average rainfall per day.
- **Adding or substracting**: e.g., adding daytime and nighttime temperatures might be useful to assess average thermal conditions.
- **Interaction terms**: e.g., polynomial features, to capture non-linear relationships. For example, squaring wind speed, since stronger winds have a disproportionate errosion effect.

Dimensionality Reduction

- **Dimensionality reduction**: transform data into lower dimensional space while retaining as much information as possible (usually data variance).
- **Principal component analysis (PCA)**: transforms data into a smaller set of **uncorrelated variables**, principal components. The goal is to capture as much data variance as possible with less features. E.g., reduce number of spectral bands to simplify data. The algorithm is as follows:
  1. **Standardize the data**.
  2. **Compute covariance matrix**: how features change with each other.
  3. **Calculate covariance matrix eigenvectors and eigenvalues**: eigenvectors represent the **direction of maximum variance of data**, while eigenvalues tell us **how much variance** each eigenvector explains.
  4. **Select principal components**: rank eigenvectors by their eigenvalues from highest to lowest, choose top principal components with highest eigenvalues.
  5. **Project the data**: project the original data onto the selected principal components.



Eigenvectors and eigenvalues in Principal Component Analysis (PCA).

Feature Extraction

- **Feature extraction** focuses on **creating entirely new representations** from the raw data.
- **Spatial summary feature**: aggregates information from a spatial area to produce a single metric for a location (e.g., average rainfall within a spatial grid).
  - **Zonal statistics features**: summarize values within defined spatial zones, e.g., mean, median, std, sum.
  - **Spatial density features**: capture the density of objects or events within an area.
  - **Spatial autocorrelation features**: capture how similar values are spatially clustered or dispersed, like Moran's I, e.g., clustering of high-pollution areas.
  - **Edge and boundary features**: focus on properties of boundaries or transitions between different spatial areas, such as land cover edges or coastlines.
  - **Texture features**: summarize the spatial patterns of pixel values within an area, capturing roughness, smoothness, variability. E.g., land cover types, like difference in pixel values (texture) between water bodies and forests.
- **Proximity feature**: measures the distance or closeness of a location to specific landmarks or points of interest. The difference is that spatial summary features describe the properties of an area around a point, while proximity features measure the closeness of a point to specific, targeted locations.
  - **Distance-based features**: measure straight-line or network distance from one point to another.
  - **K-nearest neighbor features**: identify closest k location or objects (e.g., cities, pollutant sources) and summarize distances or properties of these neighbors.
  - **Buffer-based features**: buffering is creating a zone around a point or area to calculate or summarize attributes within a specific proximity.
  - **Gravity model-based features**: measure the "attractiveness" or influence of one location over another based on distance, properties (population, size).

Feature Transformation

Feature transformation is applying formulas to raw data, like **logarithmic transformations** (reduce the impact of large values), **square root transformations** (stabilize variance), **scaling** and **normalization** (putting all features on the same scale).

Coordinate Encoding and Normalization

- Coordinate encoding and normalization transform raw latitude and longitude values into features that are more interpretable and effective for machine learning models.
- **Cyclic encoding**: since latitude and longitude are circular in nature ($-180 = 180$), we transform values using sine and cosine transformations. Each coordinate is transformed into two separate values, sin and cosine:
  - $\text{lat\_sin} = \sin\left(\frac{\pi \times \text{lat}}{180}\right)$
  - $\text{lat\_cos} = \cos\left(\frac{\pi \times \text{lat}}{180}\right)$
  - $\text{long\_sin} = \sin\left(\frac{\pi \times \text{long}}{180}\right)$

  - long_cos = $\cos\left(\frac{\pi \times \text{long}}{180}\right)$
- **Scaling and normalization**: latitude and longitude are on different scales (-90 to 90 and -180 and 180). We can **normalize** them to $0-1$ scale or standardize them to have 0 mean 1 variance"
  - lat_scaled = $\frac{\text{lat}+90}{180}$
  - long_scaled = $\frac{\text{long}+180}{360}$

Encoding Caterogical Variables

Encoding categorical features is the process of coverting non-numerical data into numerical format. Common method is to use **one-hot** encoding and **label encoding** (assigning unique interger value for each category).

Feature Binning

Feature binning is the process of **diving continuous data into discrete intervals**. With this we can **manage outliers** and create categorical levels from numerical data.

10: Physics-Informed ML

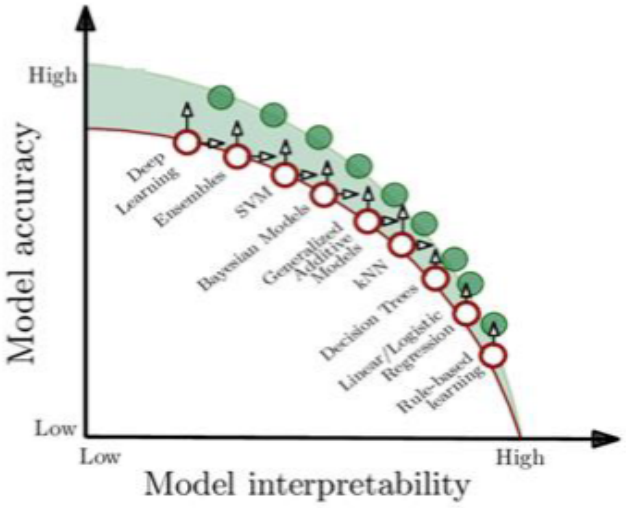
Introducing physical equations to NNs is done to prevent predictions that violate fundamental principles. One challenge in EDA is data scarcity, which forces models to extrapolate beyond the training dataset space. Another challenge is is the need for physical realism.

Physics-Informed NNs (PINNs)

PINN is a framework that integrates physical information into ML models. PINNs learn a mapping between input and output data, but with an added constraint that the NN must satisfy the underlying physical laws of the system studied.

Default NNs have these constraints that PINNs try to solve:

1. In supervised setting, NNs require lots of data.
2. NNs struggle to generalize to parts of the input space which aren't represented in the training set.
3. Challenge of overfitting in presence of noise.
4. Are considered black boxes due to the opacity of their functioning.
5. May yeild outputs that are physically illogical.



Model interoperability vs Model accuracy.

PINNs require **physical understanding of the system** and **significance of physical information integration** (choice of the method and how to incorporate physical information into a PINN).

When Not to Use PINNs

- If our understanding of the system is limited, unreliable.
- If the problem at hand is not physical in nature.
- If the problem is low-dimensional and has a relatively simple alternative model.
- If data is too limited. PINNs still require some data for training, though less than default NNs.
- PINNs are computationally expensive.

Representations of Physical Knowledge

1. **Equation-based representations**: **strict, well-defined mathematical equations** that are derived from physical laws. They tightly constrain the model, thus introducing **strong bias** in learning. They include:
   - **Algebraic equations** (force = mass x acceleration).
   - **Ordinarry differential equations (ODEs)**, describe the system evolving over time.
   - **Partial differential equations (PDEs)**, govern spatial and temporal dynamics.
   - **Stochastic differential equations (SDEs)**, include random or probabilistic components.
   - **Integral equations**, relate function to their integrals.
2. **Concept-based representations**: focus on **general, qualitative principles of physics**, rather than exact equations. They provide **weak bias**. Embedding general principles can improve model perfomance and interpretability while not constraining the model too much. Examples:
   - **Symmetry constraints**, e.g., ensuring invariance under rotation or translation.
   - **Object permanence**, the principle that objects continue to exist even when they are not directly observable.

PINNs and NN Architecture

- Any NN can be adapted to PINN. Usually used NNs are MLPs, CNNs (images, grids, spatially structured data), RNNs (time-evolving).
- PINNs use same backpropagation, optimizers.

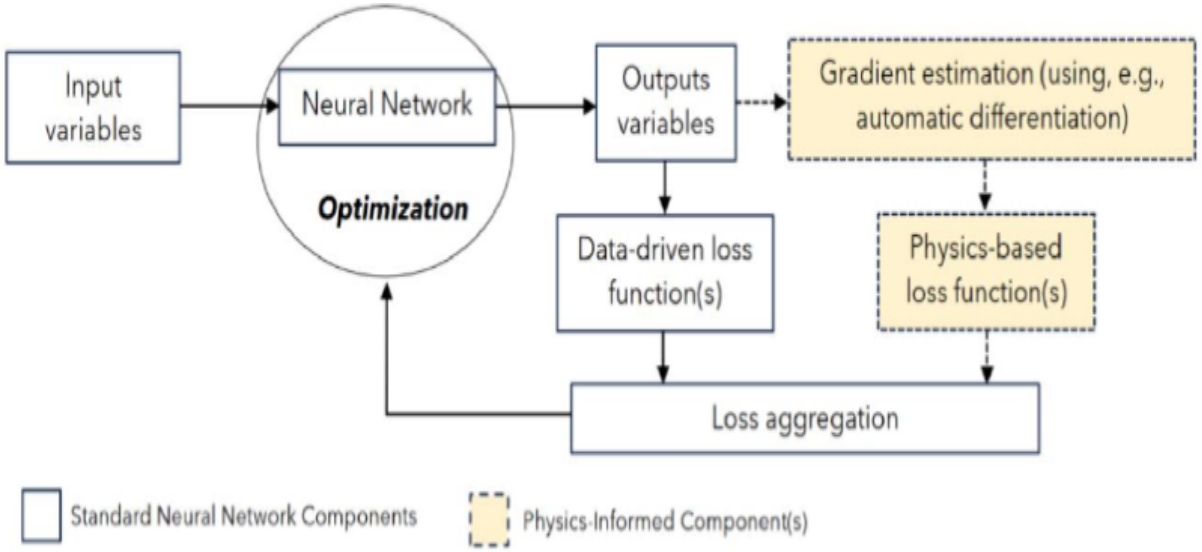
Incorporating Physics into NNs

1. **Pre-training physics integration**: used to manipulate and prepare training data.
2. **In-training physics integration**: infuse physics into training process.
3. **Architecture-level physics embedding**: modify NN's architecture or parameters to encode physical principles.

Physics-based Loss Functions

- The most common approach to create PINNs is to embed physical equations directly into the loss function (**in-training**).
- Rather than focusing solely on minimizing physical loss terms, the physical loss can acts as a **regularizer** to the primary data loss. The main objective remains data fitting, the physics loss ensures the learned function doesn't deviate significantly from physical laws. Done as either **soft** or **hard constraints**.
- There are various type of estimating **derivatives** in the physical loss. One common way is to use **automatic differentiation**. Another is **numerical differentiation**.

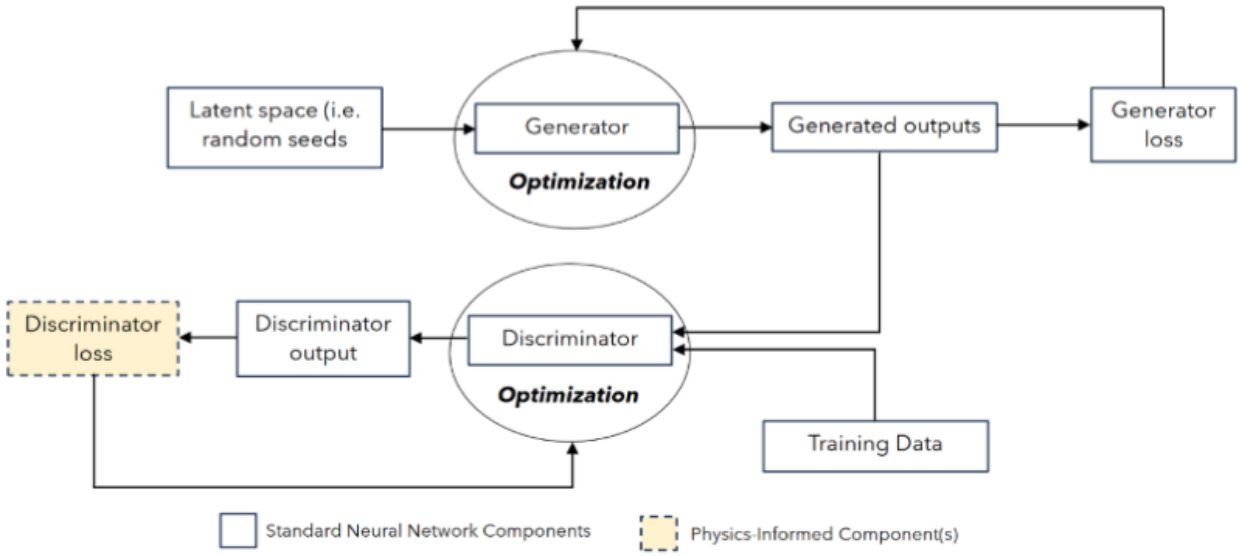
$$L_{\text{total}} = \alpha L_{\text{data\_driven}} + \beta L_{\text{physics\_informed}}$$



Schematic of how Physics-based Loss Function are incorporated in PINNs.

Physics-Informed GANs (PIGANs)

- Physical constrains are usually used in the **discriminator**. The discriminator evaluates the outputs from the generator by assessing two main aspects: realism and physical validity. It checks whether the generated data is indistinguishable from real data (realism) and whether it conforms to the governing physical laws or constraints (physical validity).



Schematic of how Physics-informed Generative Adversarial Networks work.

Architecture-level Physics Embedding

- Incorporating physics-based concept directly into the design of a NN. Can be fixed or adjusted during the training process.
- E.g., physics-informed **activation functions**.

11: Probabilistic ML

- **Uncertainty** is inherent unpredictability, variability in natural processes, human activities, and numerous unknown aspects.
- **Error** is the difference between measured, predicted, or observed value and the true or accepted value. Without true value, we address uncertainty.

Uncertainty

1. **Aleatory uncertainty**: stochastic or irreducible uncertainty **given our understanding** of the system. It arises from inherent randomness or variability of a system. Think of weather variability and chaotic nature of weather systems.
2. **Epistemic uncertainty**: reducible or knowledge-based uncertainty, it arises from a lack of understanding, incomplete information, or insufficient data. It can be reduced by improving models, colleting more data, or gaining better insights into the underlying processes.

Note that the boundary between aleatory and epistemic uncertainties can be blurry - what today seems random may later be understood as predictable. This is why aleatory uncertainty might be considered as a subset of epistemic uncertainty. Even with flipping coin, with precise knowledge of the coin's weight, spin velocity, and air resistance, we might predict the result.

Classification of Uncertainty

1. **Data uncertainty**: originates from input data used for analysis or modeling. It arises due to errors or limitations in the instruments or sensors used to collect data, missing data, or inconsistencies in datasets.
2. **Representativeness uncertainty**: arises from the inability of data to accurately represent the actual characteristics of the phenomenon it was intended to describe. For example, taking sample from just 1-2 river points to understand water characteristics.
3. **Structural uncertainty**: results from limitations or simplications in the conceptual framework of a model or analysis. E.g. neglecting certain key input features, assuming non-linear system as linear.
4. **Parameter uncertainity**: refers to uncertainty in the values of parameters used in models due to limited data, natural variability, or inaccuracies. Unlike data uncertainty, parameters are not directly measured but inferred from expert knowledge, literature, or general understanding.
5. **Scenario uncertainty**: reflects the uncertainty in selecting or assuming future scenarios.
6. **Model uncertainty**: when simulating the past state of an environmental system, **structural, parameter and data uncertainties collectively propagate through the model** and result in model uncertainty.
7. **Predictive uncertainty**: When predicting the future state of an environmental system, factors contributing to model uncertainty, combined with scenario uncertainty, collectively propagate through the model, resulting in predictive uncertainty.

Quantifying Uncertainty

Uncertainty is measured using a range of possible values.

- **Aleatory uncertainty**: quantified using frequentist approaches or stochastic modeling, where probabilities represent the long-term frequences of outcomes under repeated trials.
- **Epistemic uncertainty**: addressed using Bayesian methods. In Bayesian inference, probabilities represent degrees of belief or confidence about unknown quantities (e.g. model parameters) and are updated as more data becomes available.
- **Fuzzy logic**: fuzzy logic represents uncertainty through degrees of membership in fuzzy sets rather than probability distributions, making it particularly suitable for handling imprecise or ambiguous information, such as linguistic descriptions ("high temperature" or "low risk"). There even exist **fuzzy neural networks (FNNs)**, which represent uncertainty through **membership functions** and **fuzzy rules** rather than probability distributions.

Quantifying Uncertainty Through Statistical Analysis

- **When true values are known**: quantify uncertainty by comparing observed values to true values. E.g., satellite measured land surface temperature can be compared with ground-truth (field) measurements to calculate errors. The process is:
  1. Compute the error difference between the obvserved values and true values.
  2. Evaluate the residuals to derive a distribution of errors, which reflects the uncertainty in the observed data.
- **When true values are non-available**: in this case statistical analysis can capture only **variability**, which is often used as a proxy for uncertainty. Variability reflects inherent randomness (**aleatory uncertainty**) but may fail to account for systematic biases (**epistemic uncertainty**). **Bootstraping** is used to approximate uncertainty when true values are unavailable. The process for bootstraping is:
  1. Start with data containing n observations.
  2. Randomly draw samples **with replacement** from the original dataset to create resampled dataset of the same size. E.g., $X = \{10, 15, 12, 9, 13\} \rightarrow X' = \{15, 12, 12, 13, 10\}$.
  3. Generate many such datasets to simulate variability in the data.
  4. Compute statistic of interest for each resampled dataset.
  5. Analyze the distribution of the calculated statistics to estimate the uncertainity in the statistic.

Uncertainty Propagation Analysis

Uncertainty propagation analysis (UPA) is the process of analyzing how previously quantified input uncertainty affets the outputs in a quantitative way. UPA involves **deterministic models** that do not inherently account for uncertainty. The analysis involves measuring how **variations in the input parameters influence the models' predictions**. By far **Monte Carlo simulation (MCS)** is most widely used for UPA. The process is as follows:

1. Define probability distributions for uncertain input variables.
2. Generate many random samples from these input distributions and feed them into model.
3. Compute the output for each sample, thus creating a distribution of outputs.
4. Distribution reflects how input uncertainties propagate through the model.
5. In ML, we can sample from input distribution many times and make a trained model predict on these samples to understand model uncertainty.

Probabilistic Models

Probabilistic models inherently account for uncertainty by **representing inputs, outputs and model parameters as probability distributions** rather than fixed values.

Probabilistic ML

Probabilistic ML incorporates probability theory to model and reason about uncertainty in data and predictions. This approach output **probabilities or distributions**. **Bayesian ML models** explicitly use probability theory to model uncertainty in data and parameters.

Bayesian ML Models

The key notion behind Bayesian ML models is **Bayes' theorem** which is used to update model parameters and predictions as new data becomes available. Bayesian ML models treat unknown quantities (parameters, predictions) as **random variables** and model them using probability distributions.

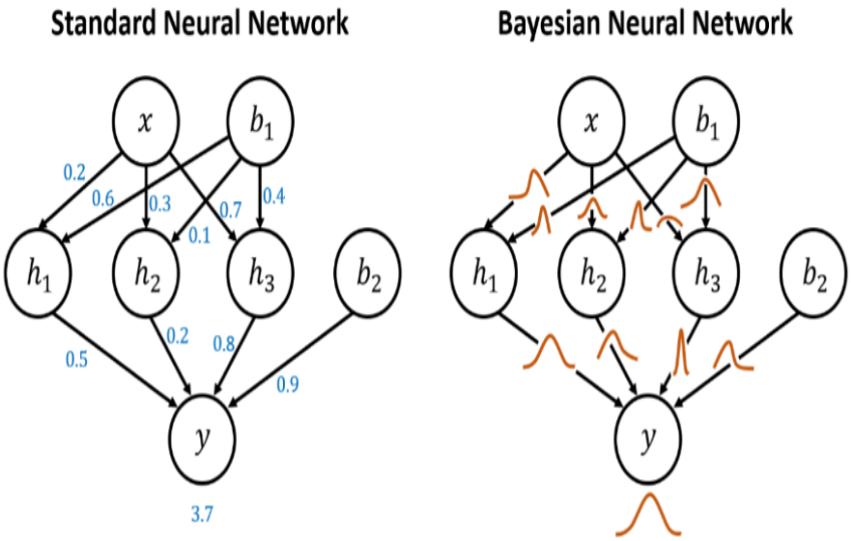
$$\text{posterior } P(\text{parameters}|\text{data}) = \frac{\text{likelihood } P(\text{data}|\text{parameters}) \cdot \text{prior } P(\text{parameters})}{\text{evidence } P(\text{data})}$$

- posterior $P(\text{parameters}|\text{data})$: updated belief about parameters after observing data.
- likelihood $P(\text{data}|\text{parameters})$: probability of observing the data given the parameters.
- prior $P(\text{parameters})$: initial belief about the parameters before observing data.
- evidence $P(\text{data})$: normalizing constant to ensure probabilities sum to 1.

These exist **Gaussian processes (GPs)**, **Bayesian NNs (BNNs)**, **Variational Autoencoders (VAEs)**.

Bayesian NNs (BNNs)

- BNNs are typical NNs, but weights and biases are modeled as probability distributions instead of fixed values.
- During inference, for a given input, multiple forward passes are performed with different sampled weights and biases. Each pass produces a slightly different prediction due to the randomness in weights and biases. These multiple predictions are aggregated to form a probabilistic output.
- During training BNNs use backpropagation, but with difference. At the start of training, **weights and biases are assigned prior probability distributions**, often Gaussian, based on assumptions or prior knowledge. Typically, all weights and biases start with the same type of prior distribution, but the parameters of these priors (e.g., mean and variance) can vary depending on domain knowledge or initialization strategy. For instance, if you expect some weights to have smaller magnitudes, you might set their prior variance to a smaller value. As the training progresses, distributions are updated using Bayes' theorem, integrating prior knowledge with the likelihood of the data. The posterior distributions of weights and biases are estimated iteratively, using **variational inference**.



Bayesian Neural Networks.

Variational Inference

Instead of directly computing posterior distribution $p(w|D)$, variational inference introduces and approximate posterior $q(w;\theta)$ (a simplified version of the posterior), with its own parameters θ (e.g., mean, variance for Gaussian). In each step of training, a set of weight and bias samples (w) is drawn from the approximate posterior $q(w;\theta)$ using the current parameters $\theta = \{\mu, \sigma^2\}$ (if Gaussian). Each sampled combination of weights and biases (w) defines a specific instance of the BNN. For each sample w, the network performs a forward pass over the training data to compute the output, and multiple forward passes (one per sample w) result in a distribution of outputs for the given input data. The **likelihood** (measures how well the sampled weights explain the data) of the data given these weights is then computed as:

$$\log P(D|w) = \sum_{i=1}^{N} \log P(y_i|x_i, w)$$

- y_i: target value (e.g., correct label).
- x_i: input data.
- w: weights and biases of the NN.
- $P(y_i|x_i, w)$: predicted probability of y_i being the correct label or value given input x_i and the specific weights w.

The likelihood $P(D|w)$ (data fit) and prior $P(w)$ together inform the objective function:

$$L(\theta) = \mathbb{E}_{q(w;\theta)}\left[\log P(D|w)\right] + \mathbb{E}_{q(w;\theta)}\left[\log P(w)\right]$$

- $\log P(D|w)$: data fit, how well the model explains the data for sampled weights w.
- $\log P(w)$: prior fit, how well the weights w match our prior beliefs about them.

Backpropagation is used to optimize θ (e.g., μ and σ^2), so that:

- $q(w;\theta)$ explains the data well $(P(D|w))$.
- $q(w;\theta)$ aligns with prior beliefs $(P(w))$.

Note that BNNs are computationally expensive to train (as multiple forward passes are required for each training step), which often limits the size of the model.

12: Generative ML

- Generative AI focuses on **generating new data** samples that **resemble a given dataset** by **learning its probability distribution**. This means the model tries to **approximate the statistical patterns and structure of the data** it is trained on. In simple terms, if a dataset is represented as a set of samples $X = \{x_1, x_2, \ldots, x_n\}$, the goal is to model the probability distribution $p(x)$ that could have produced these samples. Then we can generate new data, i.e. sample $x' \sim p(x)$ which look statistically similar to x that the model was trained on.

- Generative AI is usually **self-supervised** learning (unsupervised learning category). Self-supervised means the model derives labels itself during training, allowing it to learn more structured representations. Some models are **semi-supervised**, like **conditional GANs** (cGANs), which generate data while incorporating label information, a specific label y.
- There are **diffusion models**, **generative adversarial networks (GANs)**, **conditional GANs (cGANs)**, **variational autoencoders (VAEs)**, **energy-based models (EBMs)**, **transformer-based generative models**. They share the following characteristics:
  - Probabilistic in nature: involve modeling and sampling from a probability distribution $p(x)$, either explicitly or implicitly.
  - Rely on latent variable z, representation of input data in an abstract space or randomly sampled variable.
  - Models like conditional models incorporate additional inputs beyond latent space, ofter real or physical variables (class labels, physical parameters, external constraints). The models generate data conditioned on these inputs, enabling more targeted generation.

Generative AI in EDA

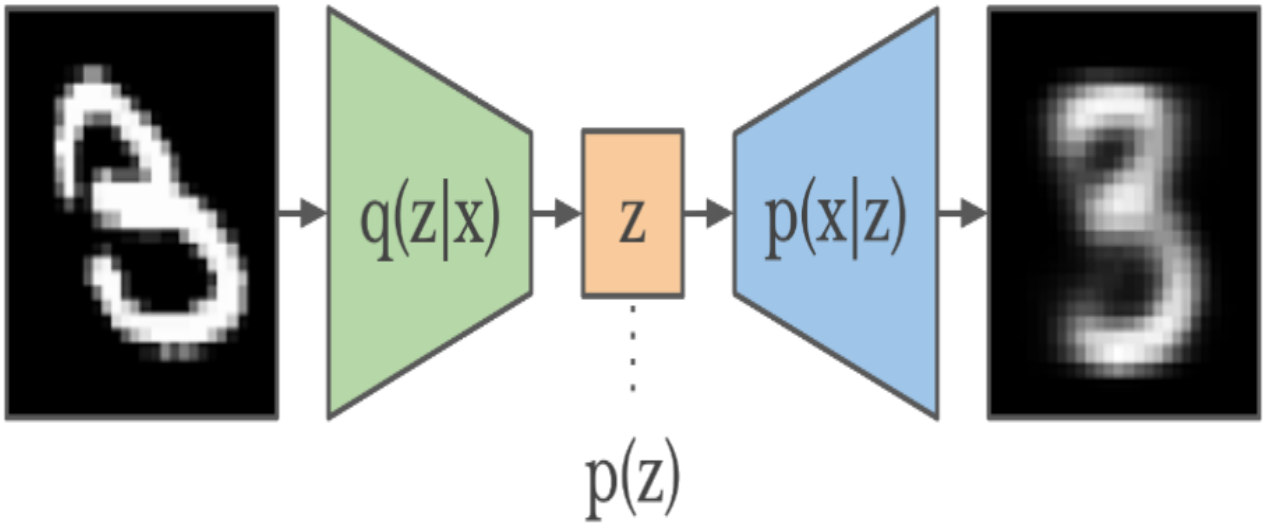
Enivonmental data suffers from data sparsity, missing information, measurement noise. With generative model we can **synthesize realistic data** to **augment datasets**:

- **Imputation**: filling gaps in time-series or spatial datasets.
- **Simulation**: generating potential environmental scenarios. **Data augmentation**: creating diverse training samples to improve robustness.

Variational Autoencoders (VAEs)

- Encoder-decoder architecture.
- Estimate the probability density of data $p(x)$.
- The encoder learns latent distribution z given data x, i.e. $p(z|x)$. By Bayes theorem: $p(z|x) = \frac{p(x|z)p(z)}{p(x)}$. $p(z)$ is the **prior distribution** over z before observing x, commonly $N(0,1)$. $p(x) = \int p(x|z)p(z)dz$, which is hard to compute.
- VAEs **approximate** $p(z|x)$ with **simpler, learnable distribution** $q(z|x)$, often modeled as a Gaussian parameterized by the encoder's output (mean and variance).
- The **encoder learns parameters** of q, such as mean and variance, by **mapping the input x to these parameters** through a neural network. The parameters of this mapping are learned by optimizing the evidence lower bound (ELBO), which serves as the objective function for both the encoder and decoder.
- The decoder maps latent variable z to $p(x|z)$, which is likelihood of reconstructing x from z. $p(x|z)$ is also modeled as a simple distribution. Decoder takes as input x and outputs the parameters of the distribution $p(x|z)$, such as mean and variance. The decoder also optimizes ELBO loss function, as ELBO depends on both encoder $q(z|x)$ and decoder $p(x|z)$.
- After training the encoder is no longer needed. z is directly sampled from prior $p(z)$ and the decoder maps z to $p(x|z)$, which generates new data x'.
- ELBO consists of:
  - **KL divergence term**: ensures that posterior $q(z|x)$ stays close to the prior $p(z)$. We minimize this term to prevent overfitting and encourage smooth and organized latent representations.
  - **Reconstruction term**: measures how well the model reconstructs x from z. We maximize this term so that the model learns to generate realistic reconstructions of x given z ensuring that z captures meaningful information about x.

In simpler terms, first encode input x into a Gaussian distribution $q(z \mid x)$, parameterized by a mean μ and standard deviation σ output by the encoder. A latent vector z is then sampled from this distribution using the reparameterization trick: $z = \mu + \sigma \cdot \epsilon$, where $\epsilon \sim \mathcal{N}(0,1)$. The decoder takes z and reconstructs x by mapping it to a distribution $p(x \mid z)$, optimizing a loss function that includes a reconstruction term (how well x is reconstructed) and a KL divergence term (regularizing $q(z \mid x)$ to be close to the prior $p(z) \sim \mathcal{N}(0,1)$). After training, new data is generated by sampling $z \sim \mathcal{N}(0,1)$ from the prior and decoding it into x'.



Role of the encoder and decode in Variational Autoencoders (VAEs)

VAEs in Point Cloud Data

There exist Point-VAE, which employs PointNet(++) encoder, which is designed to process unordered and irregular point cloud data. In addition to ELBO loss, people use **Chamfer distance** or **Earth Mover's distance** to ensure that generated point cloud data closely resembles the input cloud in terms of spatial structure.

- **Chamfer distance**: measures the average distance between points in two point clouds, ensuring that the generated cloud aligns spatially with the real cloud.
- **Earth Mover's distance (EMD)**: measures the minimal cost of transforming one point cloud into another, ensuring precise spatial matching.

Generative Adversarial Networks (GANs)

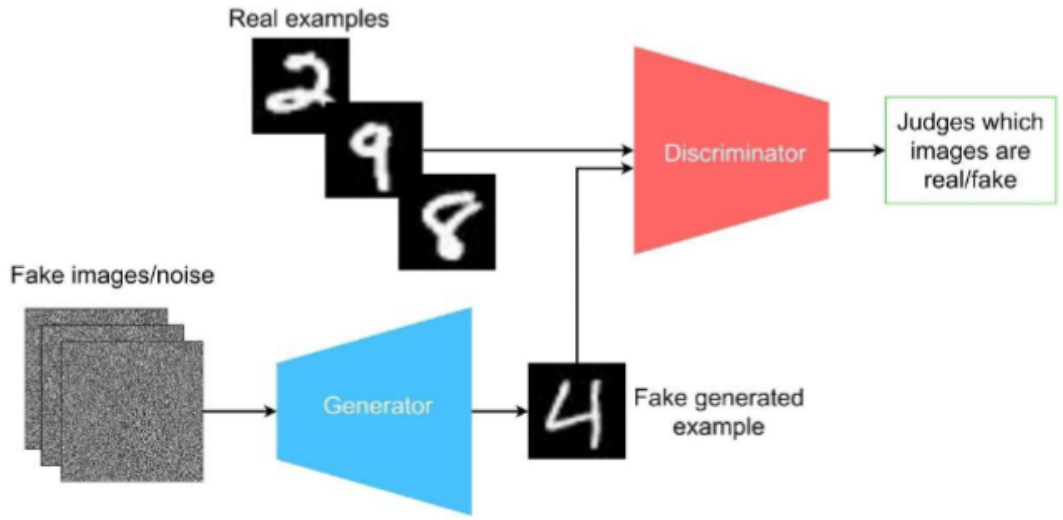
GANs use game-theoretic approach involving 2 NNs:

1. **Generator**: takes random noise z sampled from prior $p(z)$ and generates data x' that resembles real data $\sim p(x)$. It learns to map z to data distribution $p(x)$, thus trying to fool the discriminator.
2. **Discriminator**: takes real x and generated x' and outputs probability that the input is real. It learns to distinguish between real and generated data:
   - For real data x: $D(x) \to 1$ (classify as real).
   - For fake data x': $D(x') \to 0$ (classify as fake).

Adversarial loss function:

$$\min_G \max_D \mathbb{E}_{x\sim p_{data}}[\log(D(x))] + \mathbb{E}_{z\sim p(z)}[\log(1 - D(G(z)))]$$

- The **discriminator** (D) tries to **maximize the adversarial loss**, improving its ability to **distinguish real data from fake data**.
- The **generator** (G) tries to **minimize the adversarial loss**, attempting to **fool the discriminator into classifying fake data as real**.
- Ideally, the adversarial loss stabilizes at around $2\log 2$ when GAN reaches equilibrium, and distriminator output $D(G(z))$ equal 0.5 for all samples.



Schematic overview of Generative Adversarial Networks (GANs)

GANs and Point Cloud Data

- Use PointNet(++) decoders for the generator.
- PointNet(++) for the discriminator.
- **Point cloud-specific loss functions** (Chamfer, Earth Mover's distances) are often incorporated in a multi-loss framework to enhance the quality of generated data.

Diffusion Models

Diffusion model iteratively denoise a noisy signal. They rely on:

- **Forward process (diffusion)**: **progressively adds noise to data** x_0 over a series of time steps t, **producing a sequence of increasingly noisy data** x_t, until the data becomes nearly indistinguishable from random noise: $x_0 \to x_T, p_T(x_T) \sim N(0,1)$. **It does not involve any NNs or learning**. It is entirely predefined and **governed by a noise schedule**, e.g. Gaussian noise with increasing variance over time.
- **Reverse process (denoising)**: **learnable component** of diffusion models. NNs **learns to gradually remove noise** from x_t to reconstruct the original x_0: $x_T \to x_0$.
- A NN $\epsilon_\theta(x_t, t)$ is trained to predict the noise ϵ added at each step t. By accurately prediction noise, the NN iteratively removes noise to reconstruct x_0.
- The model is trained to minimize the difference between actual noise added during the forward process and the noise predicted by the denoising network.

$$\mathcal{L} = \mathbb{E}_{x_t,t,\epsilon}\left[||\epsilon - \epsilon_{theta}(x_t, t)||^2\right]$$

Diffusion Models and Point Cloud Data

- Use PointNet(++) to extract global features or graph-based networks (e.g., Dynamic Graph CNNs) to capture relationships between points for the denoising network.
- **Multiloss** is often used, combining:
  1. **Noise prediciton loss**: ensures model can accurately predict and remove noise.
  2. **Chamfer distance or EMD**: ensures th generated point cloud resembles the real data spatially.
  3. **Regularization terms**: ensure a smooth and consistent latent space or handle additional features.

13: Automated ML

- **Automated Machine Learning (AutoML)** refers to systems or tools designed to handle the entire machine learning pipeline automatically, from raw data input to producing a ready-to-deploy model. The goal is to reduce the manual effort and expertise required for trial-and-error processes in building machine learning models.

- AutoML focuses on **optimizing resources**, **minimizing the need for extensive trials**, **avoiding brute-force search**, since many configurations are redundant or suboptimal and brute-force exploration is wasteful.
- AutoML systems are **complex services** that are typically **developed** with significant effort, often **by organizations or researchers**, and made available to others either as free, **open-source tools** or as **paid services**.
- AutoML is a complex tool, because the process should work for **diverse datasets**, **ML tasks**, **numerous decisions** and **remain computationally efficient and manageable** throughout the workflow **automatically**.
- Traditional ML training workflow VS AutoML workflow:
  - Traditional: Define problem → Collect data → Preprocess data → Train model → Evaluate.
  - AutoML: Define problem → Collect data → AutoML.

AutoML for EDA

- **Challenges in environmental data and ML tasks**: environmental data is often highly complex, requiring advanced feature engineering and sophisticated model architectures.
- **Bridging the expertise gap**: there is a limited overlap between env. experts and ML specialists. AutoML enables less experienced in ML env. experts carry out advanced ML tasks.

Automate Feature Engineering

Automated feature engineering is the process of using algorithms to **automatically create**, **transform**, and **select features from raw data** for machine learning models. It involves 3 pillars:

1. **Algorithmically synthesizing new features** to augment the raw set of features.
   - Predefined list of transformations (wasteful in time and resources):
     1. **Abstraction**: **transforming individual features** to extract **new insights** or **improve usability**. Examples: **scaling**, **log transformations**, **categorical encoding**.
     2. **Combination**: **generating new features** by **combining existing ones**. Examples: **feature interactions** (products, ratios), **aggregations** (avg, min, max), **polynomial features**, **distance measures** (distances between points in multidimensional or geospatial data).
   - Dynamic methods:
     1. Generate a **customized list of transformations** on the **specific dataset**.
     2. Create **new features sequentially**, where the **results from** analyzing **one feature guide the choice of the next** feature to generate (RL, greedy forward selection, iterative feature refinement, genetic algorithm).
2. **Metric to evaluate and compare feature sets** to assess the quality, relevance, and importance of features or feature sets with respect to the target variable. This is used to discard irrelevant, redundant features.
   1. **Statistical metrics**: assess the direct relationship. Examples: **correlation coefficient** (linear), **mutual information** (non-linear).
   2. **Model-based metrics**: classical ML to assess feature importance. Examples: **tree-based models** (RF, Gradient Boosted Trees).
   3. **Predictive metrics**: directly measure effect of feature sets on the perfomance of the model. These methods are computationally expensive, since we have to train the model. Examples: **cross-validation score**, **adjusted** R^2 (regression), AUC-ROC (classification).
3. **Algorithms to systematically search the feature space**, optimizing selection of features for the task. Feature space consists of **original features** and **engineered features**. Search methods include:
   - **Exhaustive search**: evaluates all possible feature combinations.
   - **Randomized search**: randomly sample features subsets.
   - **Heuristic-based search**: guide the search with rules to reduce computational cost. Examples: **greedy forward selection**, which adds features incrementally based on perfomance; **greedy backward elimination**, which removes features iteratively starting from the full set.
   - **Evolutionary algorithms**: iteratively evolve feature subsets inspired by natural selection (???). Examples: **genetic algorithms**, **particle swarm optimization**.
   - **Bayesian optimization**: builds probabilistic models to efficiently explore high-dimensional feature spaces.

Automated Model Selection

- Final chosen model must balance accuracy, efficiency, and task-specific constraints to achieve the best results. This can be addressed by **automated model selection** through **neural architecture search (NAS)** and **hyperparameter optimization (HPO)**.
- Model selection is the process of **choosing the best-performing model from a set of predefined models** or configurations. It involves evaluating multiple existing models and **comparing their performance on a given dataset**. Models are **predifined by** the **algorithm** or **practitioner**.

- **Early termination** refers to stopping the evaluation of a model or configuration before it completes the full process. It can be achieved by:
  1. **Perfomance thresholds**: stop training if perfomance metrics fail to improve after a set number of iterations.
  2. **Budget constraints**: halt model exceeding allocated resources, such as time or computational cost.
  3. **Confidence intervals**: use statistical methods to stop evaluation when the likelihood of outperforming the best model so far is low.

Neural Architecture Search (NAS)

Neural Architecture Search (NAS) is a systematic method for creating and evaluating new neural network architectures to find the best one for a given problem. It often incorporates Hyperparameter Optimization (HPO) as part of the process to fine-tune the architectures for optimal performance. NAS is highly automated, often requiring minimal manual intervention beyond defining the search space and objective. The process involves:

1. **Defining search space**: set of all possible neural network architectures to explore. Options include **number and type of layers**, **layer connections** (sequential, skip connections), **activation functions**, and more. Common method is called **cell-based**, where architectures are constructed by assembling **small, modular components** called **cells**. A cell typically represents a specific layer or group of layers with defined operations (convolutions, polling, skip connections). Once cells are defined, NAS centers on how to combine and stack these cells to form the architecture.
2. **Search strategy**: search strategy is then used to **navigate the defined space** and **identify promising architectures**. Strategies include:
   - **Reinforcement learning (RL)**: Treats the search process as a **decision-making problem** where **actions** correspond to **choosing architecture components**.
   - **Evolutionary algoritms**: simulates biological evolution by mutating and combining architectures to find better candidates.
   - **Gradient-based methods**: **utilizes gradients to optimize the architecture** directly, often leading to **faster convergence**.
3. **Evaluation metric**: An evaluation metric, such as validation accuracy, loss, or computational efficiency, guides the search process by assessing the performance of candidate architectures.

Schematic of Neural Architecture Search (NAS).

Hyperparameter Optimization (HPO)

Hyperparameter Optimization (HPO) is the process of systematically searching for the **best set of hyperparameters** for a machine learning model to **maximize its performance** on a given task. **Hyperparameters** are model parameters that are **set before training** begins and **cannot be learned** directly **from the data** (e.g., learning rate, batch size, number of layers, or regularization strength). In AutoML, HPO is integrated into a broaded pipeline, often including model selection, data preprocessing, feature engineering. Common methods for HPO are:

1. **Grid search**: systematically evaluate **all possible combinations** of hyperparameters. Computationally expensive and inefficient.

2. **Random search**: **randomly sample hyperparameter combinations** from a predefined range. Efficient, but might be inaccurate.
3. **Bayesian optimization**: uses **probabilistic surrogate model**, such as Gaussian processes, to model objective function and iteratively select promising hyperparameters. It balances exploration (trying new configurations) and exploitation (refining good ones), reducing the number of evaluations required. It is efficient, but introduced computational overhead in maintaining and updating the surrogate model.
4. **Hyperband**: combines random search with early stopping to optimize resource allocation. It identifies poor-performing hyperparameter configurations early in the process, freeing resources to focus on more promising candidates.
5. **Evolutionary algorithms**: optimize hyperparameters by simulating processes such as mutation, crossover and selection. Computationally intensive, but effective for large and complex spaces, as it can explore diverse configurations over generations.
6. **Gradient-based optimization**: utilizes gradients of hyperparameters with respect to the validation loss to optimize them directly. Cannot handle non-differentiable hyperparameters or categorical values.
7. **Reinforcement learning**: treats hyperparameter optimization as a sequential decision-making problem, where each action corresponds to choosing a set of hyperparameters. Rewards are based on validation performance, and the process evolves over multiple iterations. Effective, but expensive and complex to implement.

Meta Learning

- **Meta-learning**: concept in AutoML that leverages knowledge from previous machine learning tasks or trials to improve the efficiency and performance of future ones. Optimizes processes like model selection, hyperparameter optimization (HPO), and pipeline design by utilizing patterns and insights from prior experiments or datasets.
- **Learning from historical data**: AutoML systems **build a "meta-dataset"** by **storing information from past tasks**, such as dataset characteristics, model performance metrics, and hyperparameter outcomes. This **enables the system to predict suitable models, hyperparameters, or workflows for new datasets**.
- **Building meta-models**: meta-models are trained to **map datasets characteristics to optimal ML pipelines or configurations**. For example, if **high-dimensional datasets** previously performed **well with specific algorithm**, the system will **prioritize this algorithm for similar datasets**.
- **Cold and warm start**:
  - **Cold start**: improves perfomance for new datasets **without prior knowledge by leveraging general trends from releated tasks**.
  - **Warm start**: accelerates optimization for similar tasks by **reusing previously successful configurations or models**.

Pipeline Automation

1. **Modularity**: The workflow is divided into distinct, independent stages (e.g., data preprocessing, feature engineering, model selection). Each stage performs a specific task, and the outputs from one stage are passed as inputs to the next.
2. **Workflow generalization**: ML pipelines are designed to work across a variety of datasets and tasks with minimal customization.
   - **Parameterization**: configurable parameters allow pipelines to **adapt to different inputs**, such as scaling techniques, model types.
   - **Template pipelines**: **predefined workflows** provide a starting point that users can customize further.
3. **Pipeline orchestration**: ensures that individual components and core concepts (modularity, abstration, etc.) are coordinated effectively. It connects and manages execution of the entire workflow, handles dependencies, task scheduling, resource optimization.

AutoML Services

- **Auto-sklearn**: automates the process of selecting and tuning machine learning models using **scikit-learn's library of algorithms**. Its strength lies in its built-in **ensemble learning capabilities** and use of **Bayesian optimization** to identify the best models.
- **TPOT**: genetic algorithms to optimize machine learning pipelines. Uses **evolutionary approach** to iteratively improve pipeline perfomance.
- **H2O AutoML**: **fully automated** end-to-end machine learning **pipeline** that includes **data preprocessing, model training, hyperparameter tuning**, and **model stacking**. It's especially **powerful for large-scale enterprise applications**, thanks to its scalability and support for distributed computing.
- **PyCaret**: low-code AutoML, for **beginners** and **rapid prototyping**. It covers the entire ML pipeline.
- **Google Cloud AutoML**: scalable, cloud-based solution for ML automation. Supports diverse data types (image, text, video) and suitable for people seking customizable and production-ready models.