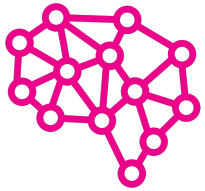


FourthBrain

MLE Program, Cohort 11 (MLE11)

Week 4: Data Engineering Basics

Agenda for Today

- Capstones Q&A (20 min)
- Data Engineering Theory (2.5 hrs)
- Break (30 min)
- Coding Assignment (~ 2.5 hrs)

Recap of Last Week!

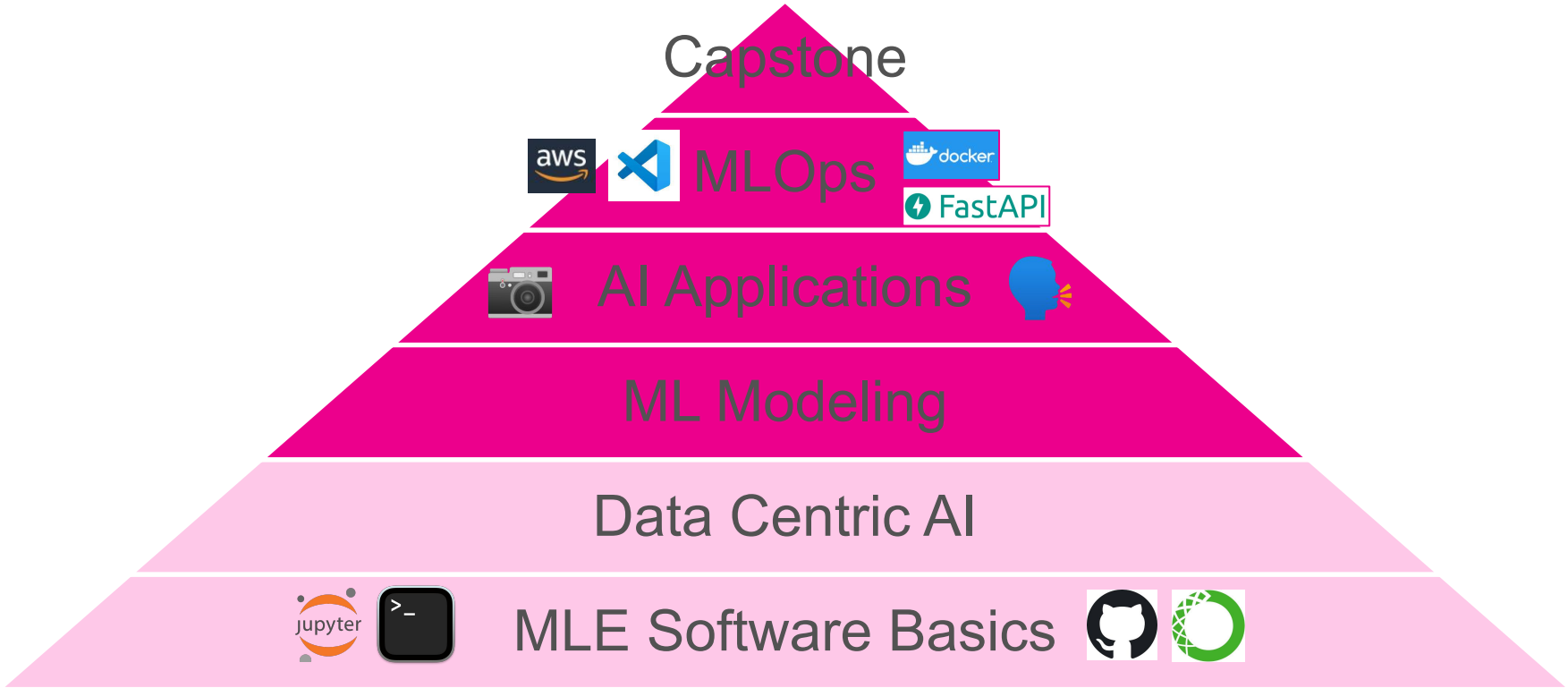
Concepts

- Best practices for high-quality data
  - Data Lineage
  - Identifying, Sourcing, Collecting, Labeling, Evaluating, and Validating Data
  - Responsible Data (ChatGPT)
- Hugging Face and FAST APIs

Hands-On Activities

- Interacting with the Hugging Face API
- Building a simple web app using Fast API
- Creating an API endpoint for your selected pre-trained model

Becoming a Machine Learning Engineer



Our Curriculum

DATA CENTRIC AI

- AI Product Development
- Data-Centric AI in the Real World
- **Data Engineering**
- Big (and Good) Data

ML MODELING

- Supervised ML
- Deep Learning & AutoML
- Unsupervised, Semi- & Self-supervised Learning

AI APPLICATIONS

- Computer Vision
- Natural Language Processing
- Transformers & Fine Tuning Pre-Trained Networks

MLOps

- ML in Production
- ML App Infrastructure
- Model Serving, Management, and Delivery
- Monitoring and Automating Pipelines

🤖 This Week!

Concepts

- Data Engineering Workflows
- Data Wrangling & Exploratory Data Analysis
- Feature Selection & Engineering
- Data Leakage
- Building Pipelines

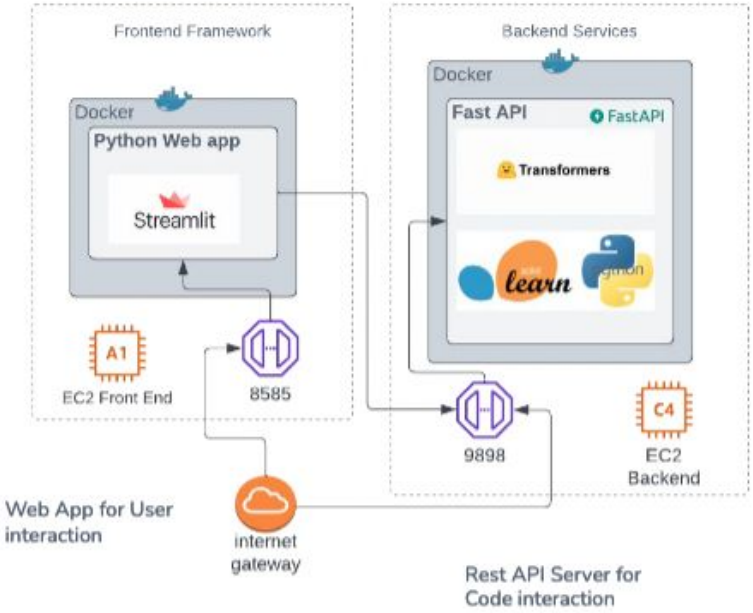
Hands-On Activities

- Exploring and wrangling structured data for sales prediction pipeline
- Building Airflow workflow

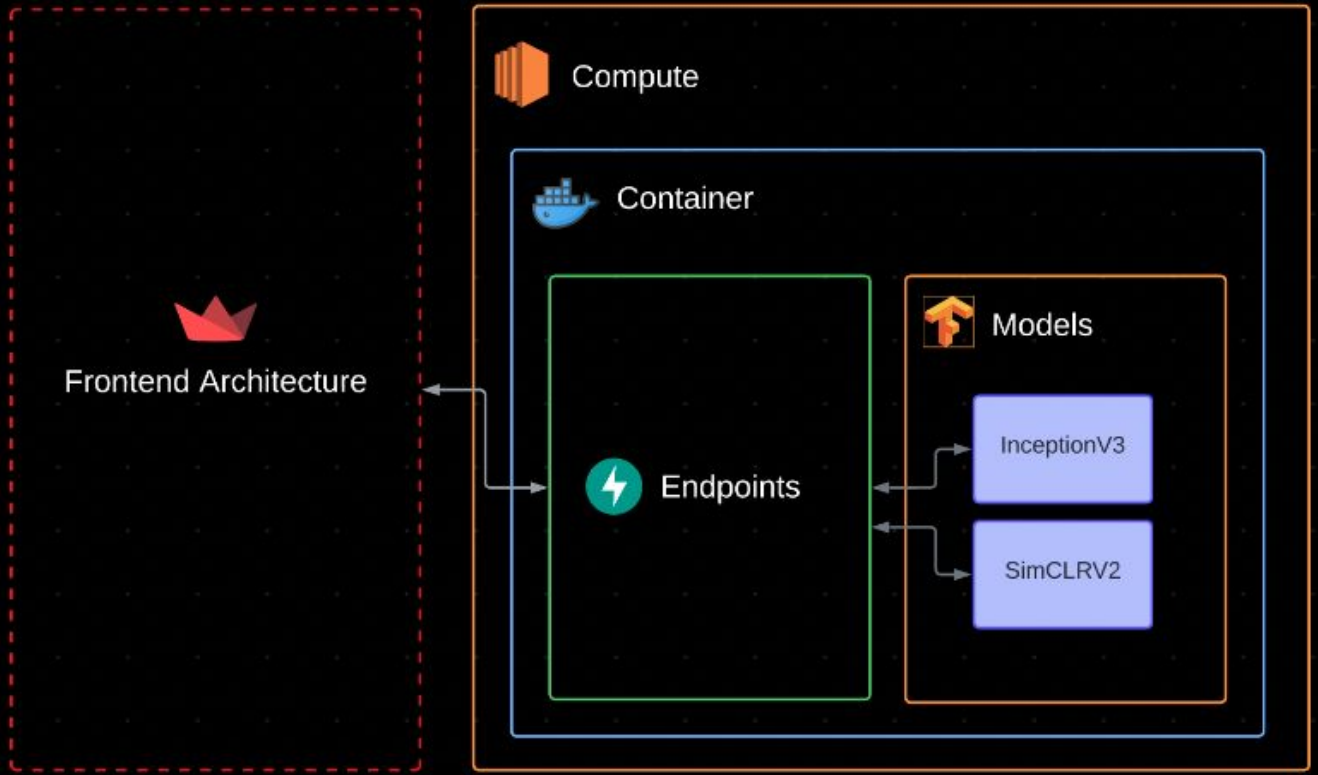
Capstone and Reminders

- **Initial Capstone Presentations in class week 5 and week 6**
  - 5 mins/team
  - Template
  - Goal - establish a plan of execution as a timeline, define clear roles and responsibilities, if you have a chance to start working on EDA - present your results, how will you deploy your model?
- Code Freeze is a thing!

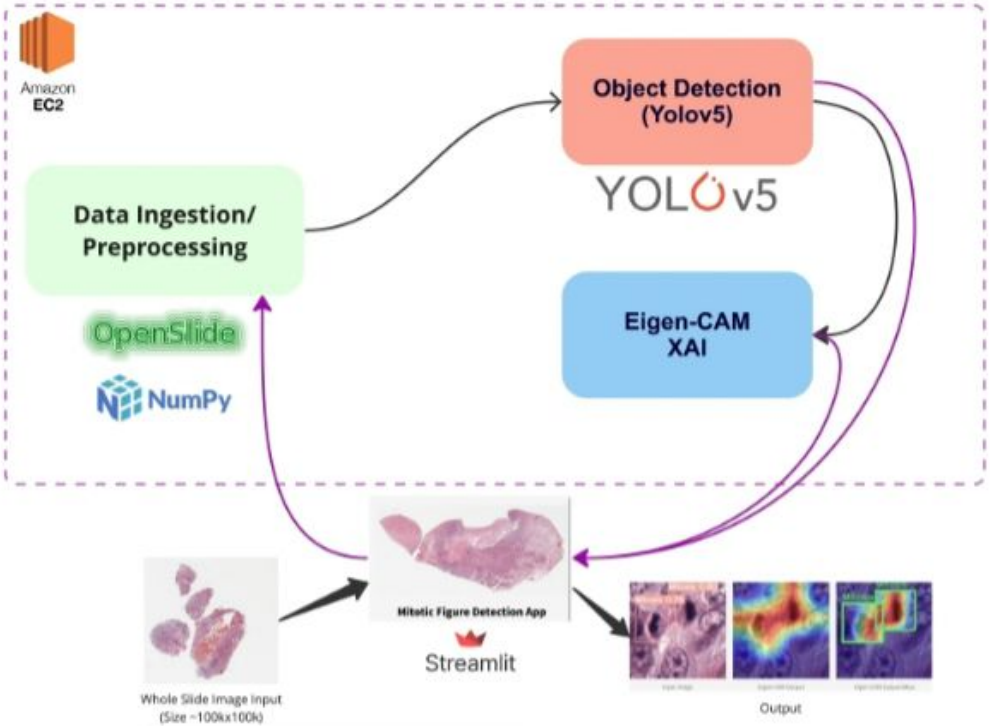
Architecture Diagram Map



SOLUTION ARCHITECTURE DIAGRAM



Web App Infrastructure



Do you have any questions?

🤖 This Week!

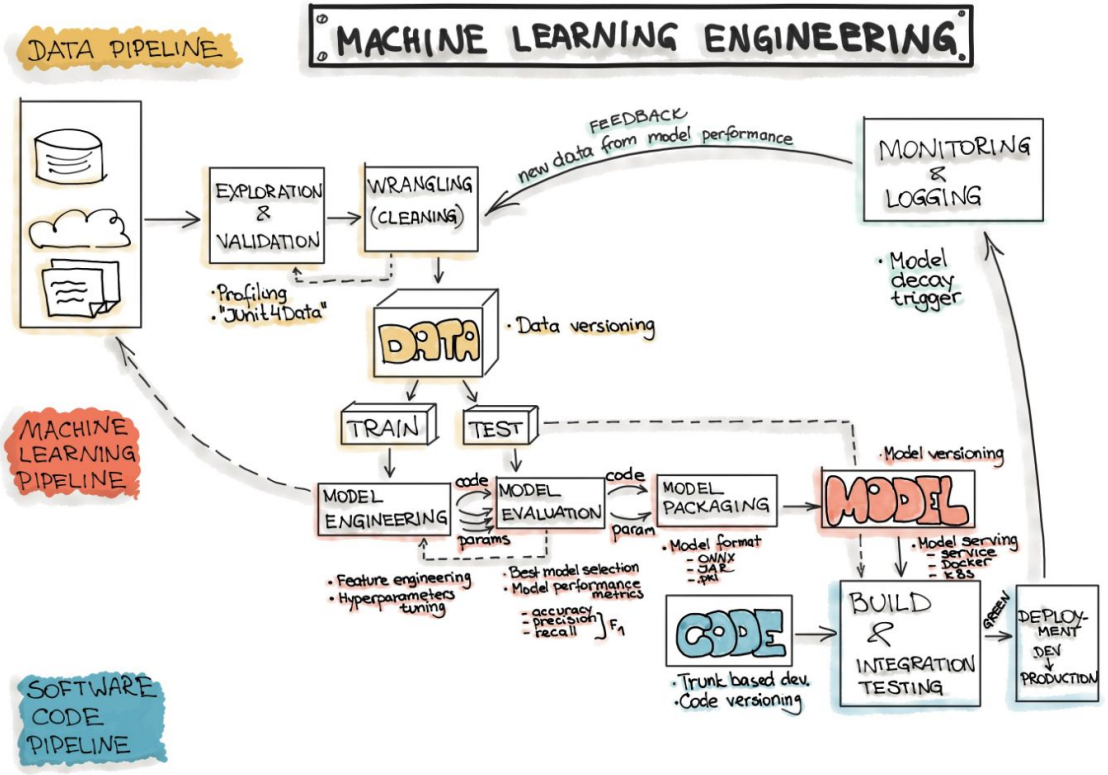
Concepts

- **Data Engineering Workflows**
- Data Wrangling & Exploratory Data Analysis
- Feature Selection & Engineering
- Data Leakage
- Building Pipelines

Hands-On Activities

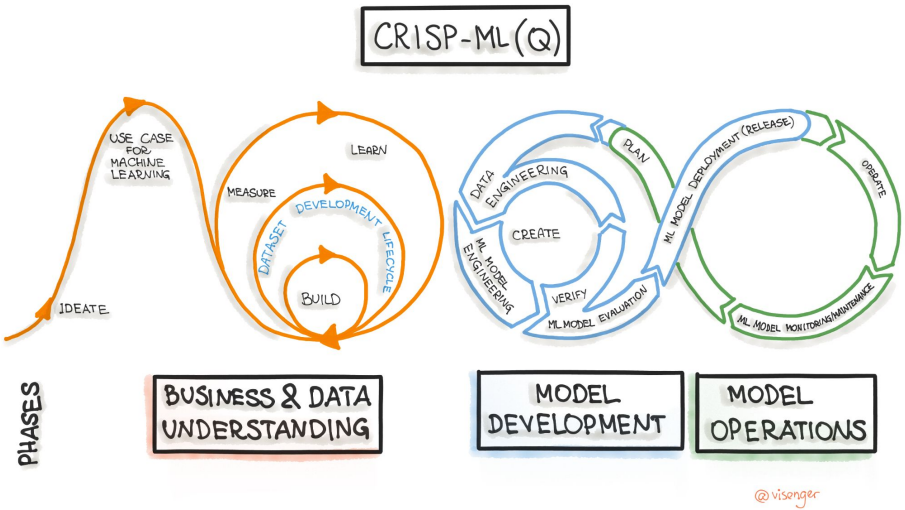
- Exploring and wrangling structured data for sales prediction pipeline
- Building Airflow workflow

End-to-End Machine Learning Workflow



Recall ... ML Applications in Six Steps

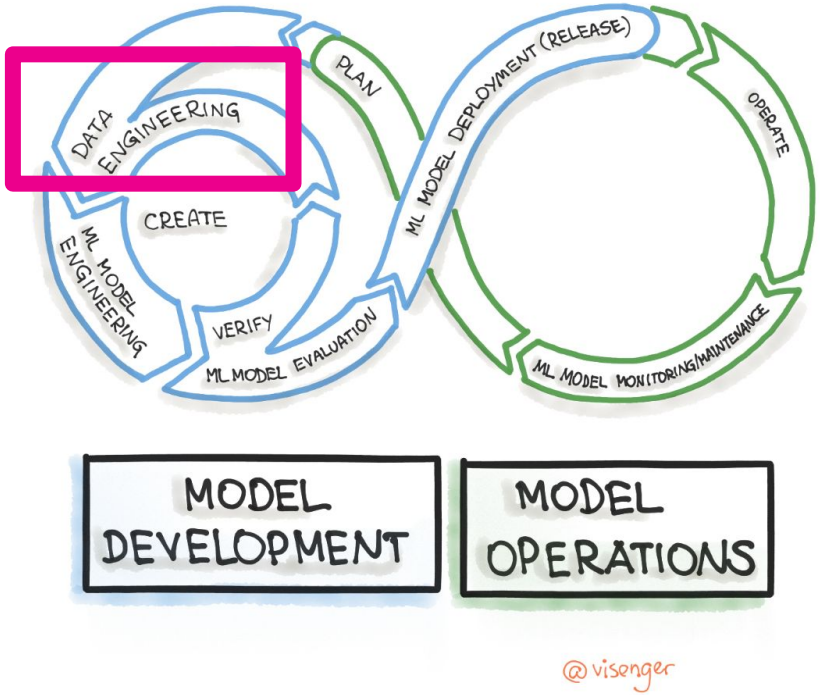
1. Business and Data Understanding
2. **Data Engineering (Data Preparation)**
3. Machine Learning Model Engineering
4. Quality Assurance for Machine Learning Applications
5. Deployment
6. Monitoring and Maintenance

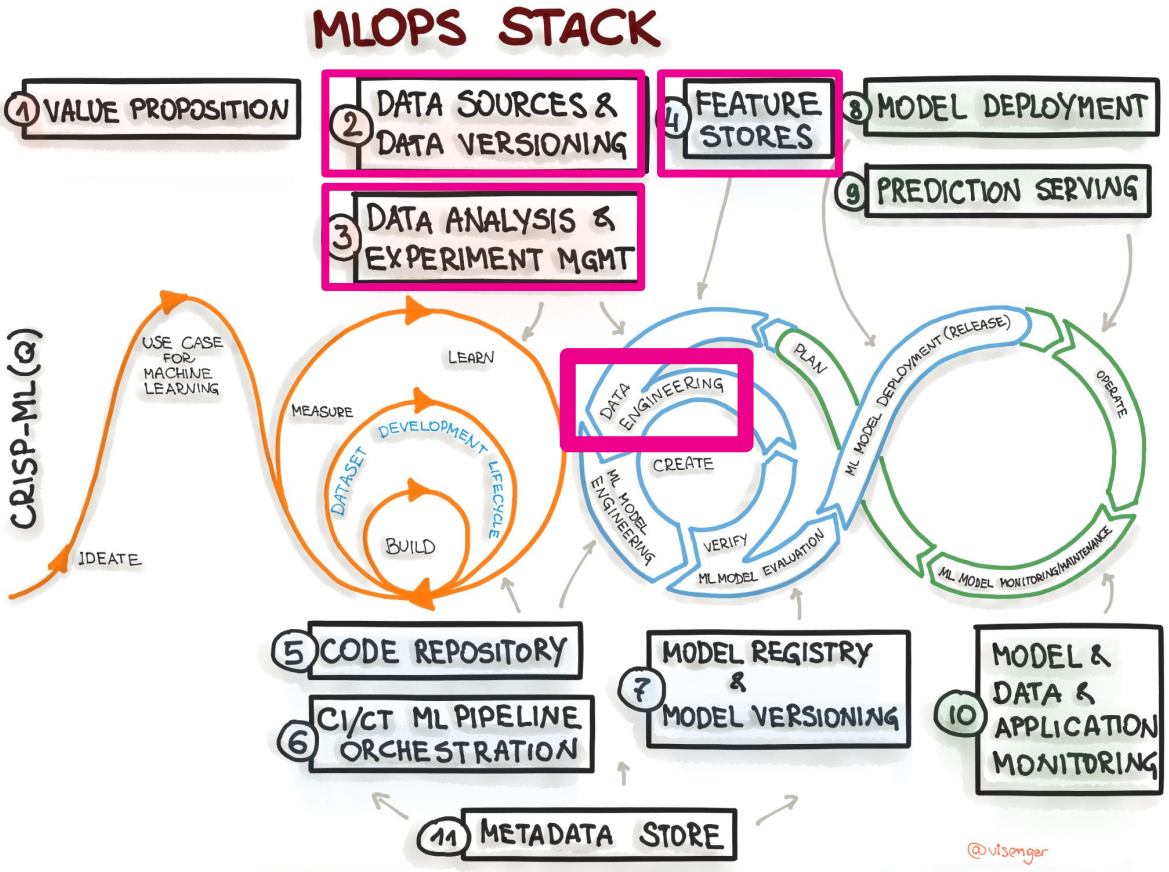


MLOps in a Nutshell

2. Data Engineering

3. ML Model Engineering
4. Quality Assurance
5. Deployment
6. Monitoring & Maintenance





🤖 This Week!

Concepts

- Data Engineering Workflows
- **Data Wrangling & Exploratory Data Analysis**
- Feature Selection & Engineering
- Data Leakage
- Building Pipelines

Hands-On Activities

- Exploring and wrangling structured data for sales prediction pipeline
- Building Airflow workflow

Data Wrangling & Exploratory Data Analysis

Role of Exploratory Data Analysis & Data Wrangling

Exploratory Data Analysis

- Perform initial investigations
- Profile the data
- Calculate statistics
- Create visualizations
- Discover patterns
- Spot anomalies and errors
- Test hypotheses
- Check assumptions
- Compute correlations between attributes and against the targets

Data Wrangling (Cleaning)

- Reformat particular attributes
- Correct errors in data
- Deal with missing values
- Fix or remove outliers
- Drop irrelevant attributes
- Restructure the data

Exploratory vs Confirmatory Data Analysis

- Exploratory Data Analysis
  - Finds a good description
  - Raise new questions

Descriptive Statistics

- Confirmatory Data Analysis
  - Tests Hypothesis
  - Settle Questions

Inferential Statistics

Descriptive vs. Inferential Statistics

- **Descriptive**: e.g., Mean; describes data you have but can't be generalized beyond that

- **Inferential**: e.g., t-test, that enable inferences about the population beyond our data
    - These are the techniques we'll leverage for Machine Learning and Prediction

Examples of Business Questions

- Simple (descriptive) Stats
  - “Who are the most profitable customers?”
- Hypothesis Testing
  - “Is there a difference in value to the company of these customers?”
- Segmentation/Classification
  - What are the common characteristics of these customers?
- Prediction
  - Will this new customer become a profitable customer? If so, how profitable?

adapted from Provost and Fawcett, “Data Science for Business”

What is EDA?

Exploratory Data Analysis (EDA) is an approach/philosophy for data analysis using a variety of techniques

Breakout

5 min
(3-4 per room)

- **What are the reasons to do EDA?**

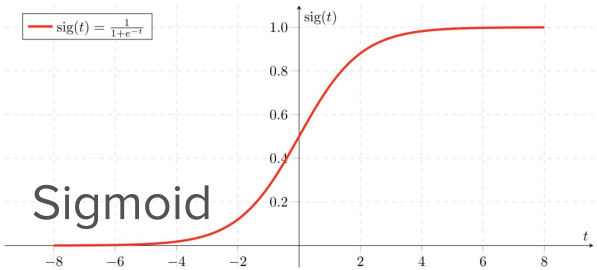
- Designate *one person to share* from your breakout room

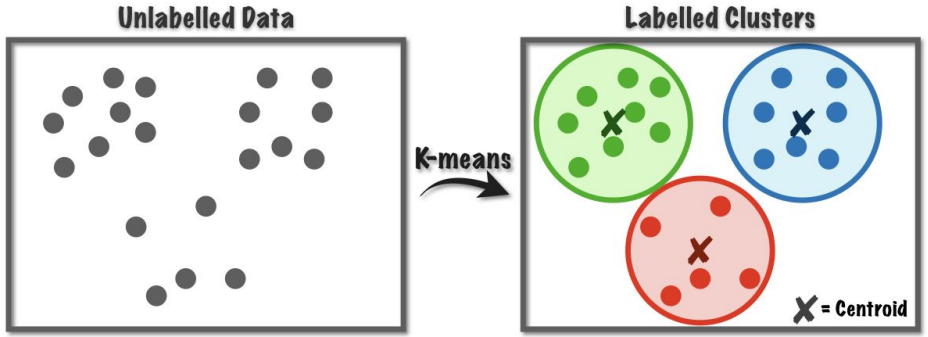
Reasons to do EDA:

- Understanding the structure and distribution of the data
- Identifying patterns and trends in the data
- Identifying potential problems or issues with the data
- Informing feature engineering
- Selecting appropriate machine learning algorithms
- Delivering data-driven insights to business stakeholders

Data Gives You Clues About The Model Selection

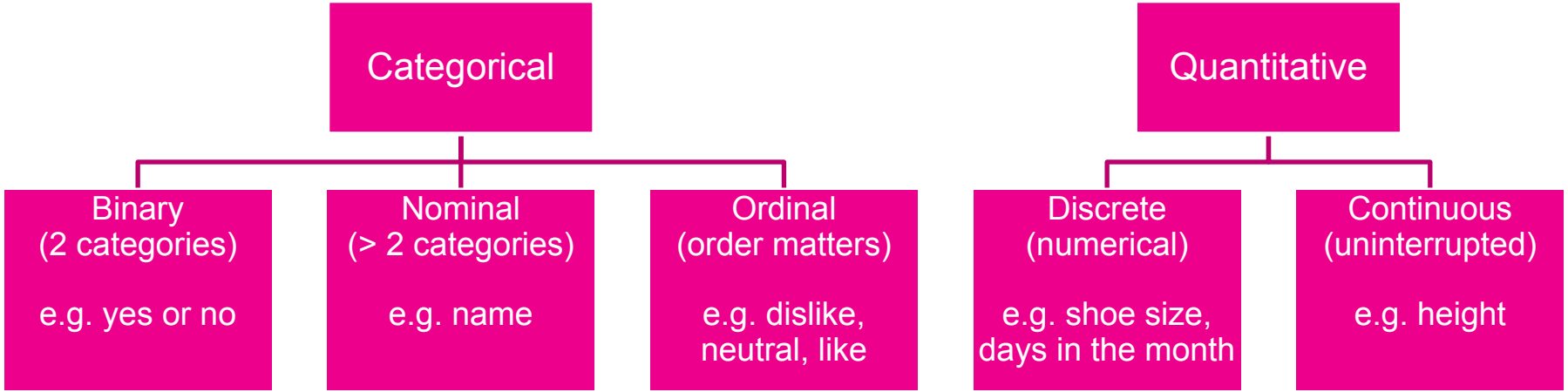
- What models/techniques to use depends on the problem context, data and underlying assumptions.
- e.g., Classification problem with binary outcome? -> logistic regression, Naïve Bayes, ...
- e.g., Classification problem but no labels?
    - -> Perhaps use K-means clustering





Jeffares, 2019

Types of Data



Two Categories of Data

- **Structured Data types**

  Example:
  - csv file, excel file, database file

- **Unstructured Data types**

  Examples:
  - text, images, videos, audio

Dimensionality of Data

- **Univariate**: Measurement made on one variable per subject

- **Bivariate**: Measurement made on two variables per subject

- **Multivariate**: Measurement made on many variables per subject

DEMO

5 min

Titanic EDA

🤖 This Week!

Concepts

- Data Engineering Workflows
- Data Wrangling & Exploratory Data Analysis
- **Feature Selection & Engineering**
- Data Leakage
- Building Pipelines

Hands-On Activities

- Exploring and wrangling structured data for sales prediction pipeline
- Building Airflow workflow

Feature Engineering

"Feature engineering is the process of transforming raw data into features that better represent the underlying problem to the predictive models, resulting in improved model accuracy on unseen data."

– Jason Brownlee

“Coming up with features is difficult, time-consuming, requires expert knowledge. 'Applied machine learning' is basically feature engineering.”

– Andrew Ng

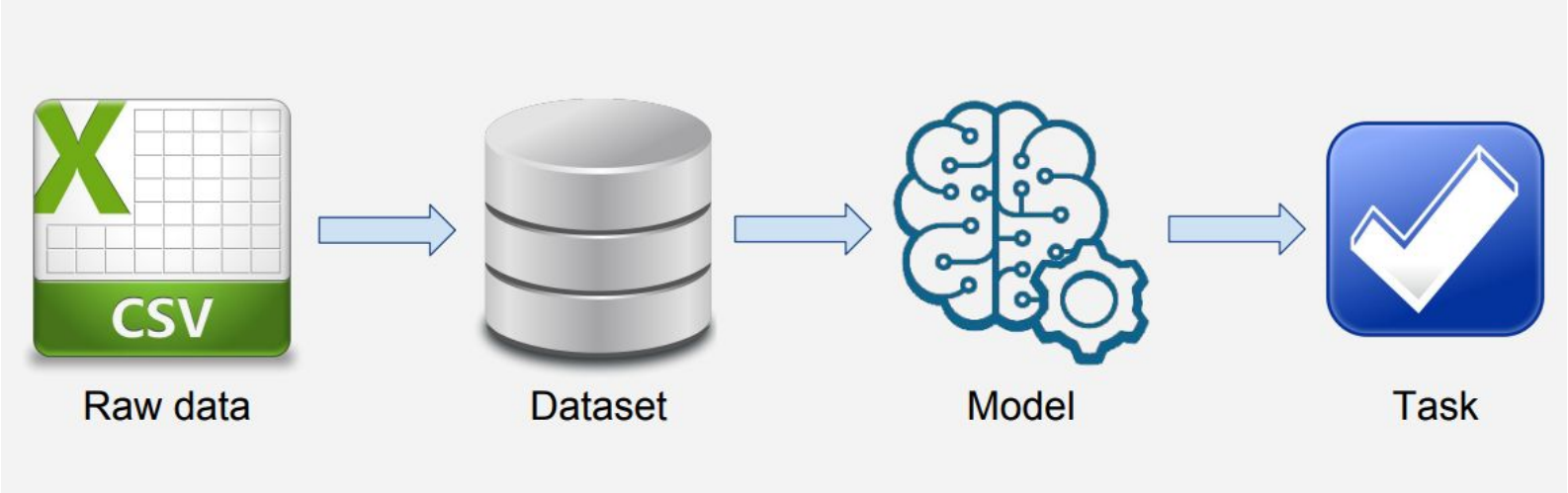
Feature Engineering Example

Imagine you have a dataset of customer data for a retails company, including information about:

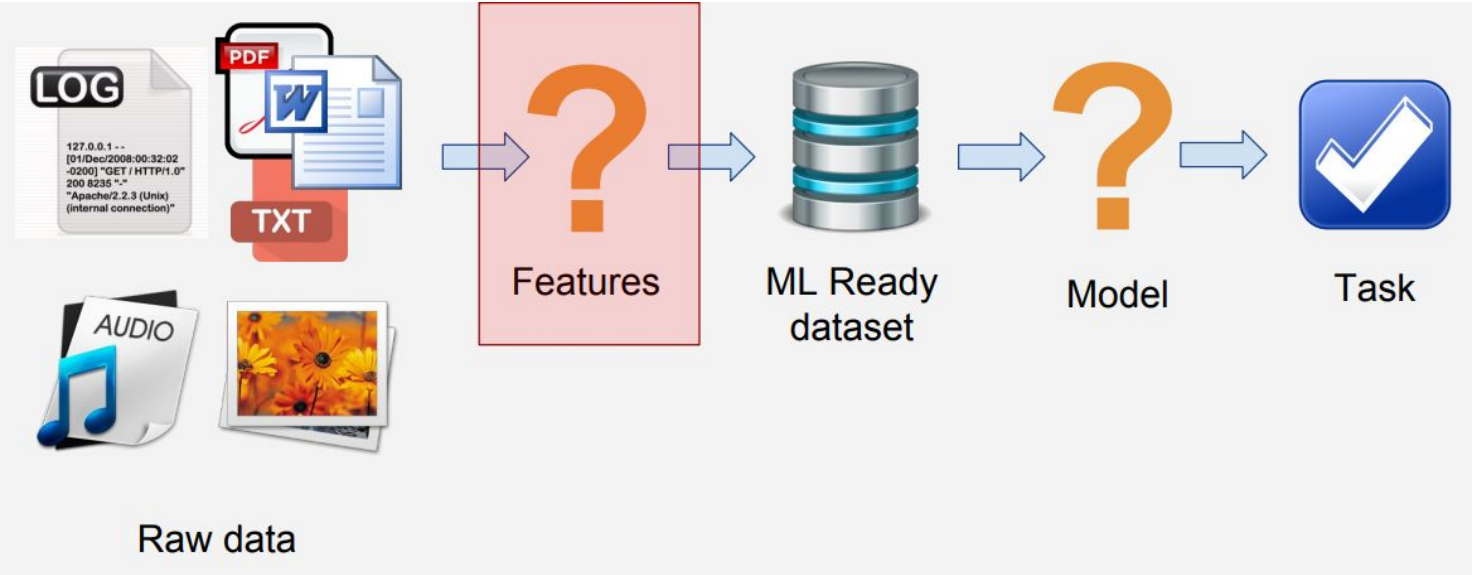
- the customer age
- income
- location
- purchase history

What potential features
could we engineer from this data?

The Dream!



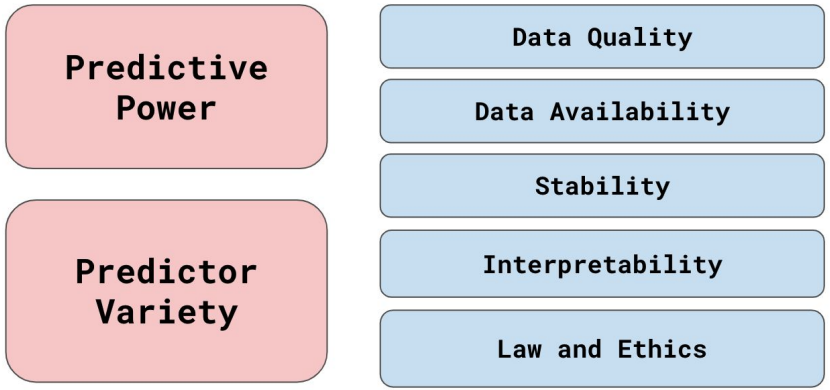
The Nightmare!



What is good Feature Engineering?

A ML model is only as good as the features it learns.

- Descriptive
- Separable
- Repeatable



Feature Engineering Process

Feature engineering consists of various process

- Define the problem
- Gather and preprocess the data
- Identify and select features
- Engineer new features
- Evaluate and refine the features
- Use the features to train the model

Tools Supporting The Feature Engineering Process

Engineering The Features:

- Domain Knowledge
- Prior Experience
- EDA

Checking Features Performance:

- Testing the model on a separate data subset
- Measurement of desired metrics (e.g. accuracy)

Feature Engineering is difficult!

- Powerful feature transformations (like target encoding) can introduce leakage when applied wrong -> more on this later today :)
- Usually requires domain knowledge about how features interact with each other
- Time-consuming, need to run many, many experiments
- Why Feature Engineering matters
    - Extract more relevant, well-performing features, remove irrelevant or noisy features
    - Simpler models with better results

Feature Engineering Techniques

Feature Engineering of numerical values - Imputation

- Datasets contain missing values, often encoded as blanks, NaNs or other placeholders
- Ignoring rows and/or columns with missing values is possible, but at the price of losing data which might be valuable
- Better strategy is to infer them from the known part of data
- Strategies:
  - Mean: Basic approach
  - Median: More robust to outliers
  - Mode: Most frequent value
  - Using a model: Can expose algorithmic bias

Feature Engineering of numerical values - Handling outliers

- Depending on the model, the effect can be large or minimal.

- The various methods of of handling outliers:
  - Removal
  - Replacing Values
  - Capping
  - Discretization

Feature Engineering of numerical values - Binarization

Transform discrete or continuous numeric features in binary features, typically by thresholding the values at a certain point

For example, suppose you have the following data:

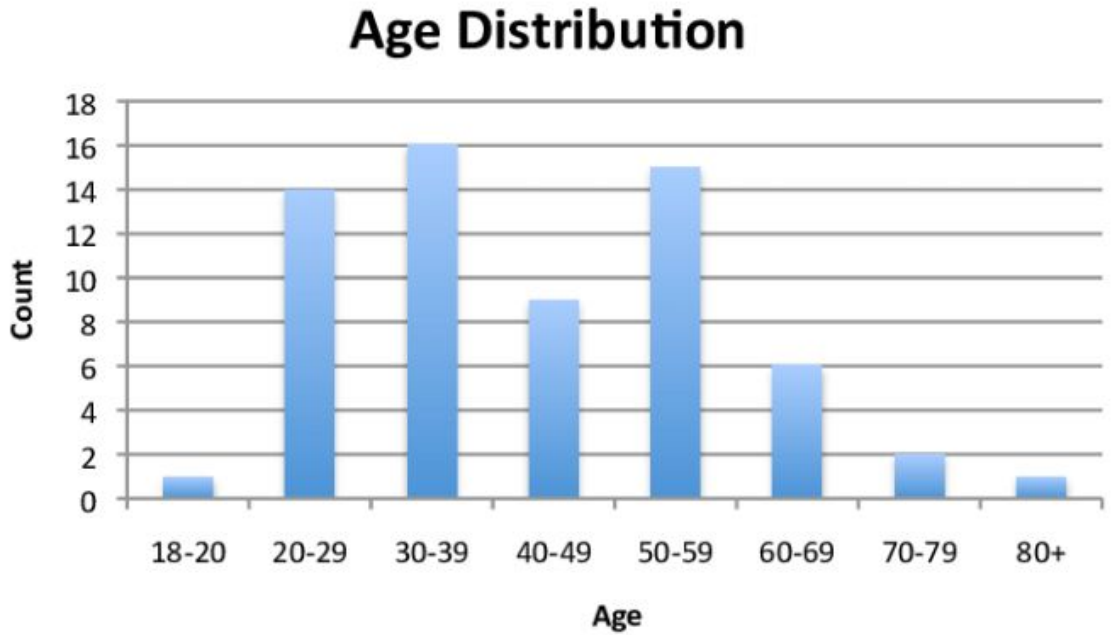
Patient	BMI	Condition
1	22	No
2	27	Yes
3	30	Yes
4	32	No

Using a threshold of 25, you could binarize the BMI feature as follows:

Patient	BMI (binarized)	Condition
1	Low	No
2	High	Yes
3	High	Yes
4	High	No

Feature Engineering of numerical values - Binning

Binning is a technique in feature engineering that involves dividing a continuous-valued feature into a set of bins or intervals, and replacing the values with the label of the appropriate bin.



Feature Engineering of numerical values – Log transformation

Involves applying the logarithmic function to a continuous-valued feature in order to transform it into a new, transformed feature

House	Size (sq ft)	Size (log transformed)	Price
1	1,000	3.0	300,000
2	2,000	3.3	400,000
3	3,000	3.5	500,000
4	4,000	3.7	600,000
5	5,000	3.8	700,000
6	6,000	4.0	800,000
7	7,000	4.1	900,000
8	8,000	4.3	1,000,000
9	9,000	4.4	1,100,000
10			

Feature Engineering of numerical values – Scaling

Feature scaling is a technique in machine learning that involves transforming the values of a feature into a common range or scale.

- Some models are sensitive to the scale of input features, e.g. neural networks, SVMs, KNN, linear regression

- Reasons to do scaling: improving model performance, handling skewed data distribution, improving model interpretability

- Popular techniques (see next slides)
    - **MinMax Scaling**
    - **Standard (Z) Scaling**

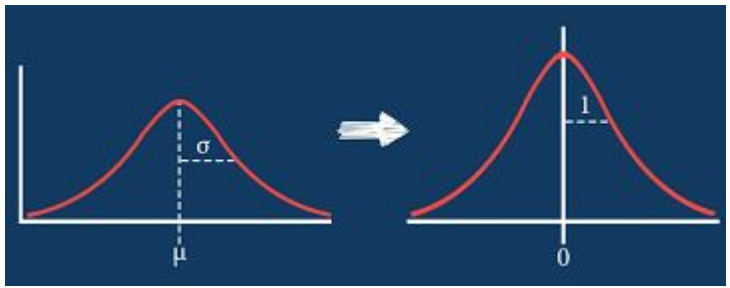
Feature Engineering of numerical values – Scaling

- **Min-Max Scaling:**

Squeezes (or stretches) all values within the range of [0, 1] to add robustness to very small standard deviations and preserving zeros for sparse data.

- **Standard Scaling:**

After standardization, a feature has mean of 0 and variance of 1 (assumption of many learning algorithms)



Geller, 2019

Feature Engineering of categorical features

- Nearly always need some treatment to be suitable for models
- High number of distinct values (High Cardinality)
- Difficult to impute missing values
- High dimentionality
- Examples of categorical features:
  - Platform: [“desktop”, “tablet”, “mobile”]
  - Document_ID or User_ID: [121545, 64845, 121545]

Feature Engineering of categorical features – Ordinal Encoding

- Each unique category value is assigned an integer value
- The integer values have a natural ordered relationship between each other and machine learning algorithms may be able to understand and harness this relationship.

Category	Small	Medium	Large
Encoded	1	2	3

Feature Engineering of categorical features – One Hot Encoding

- Creates a new binary column for each unique category in a categorical feature, with a value of 0 or 1 indicating the presence or absence of the category in a given row.

fruit	fruit_apple	fruit_banana	fruit_orange
apple	1	0	0
banana	0	1	0
orange	0	0	1
apple, banana	1	1	0

Feature Engineering of categorical features – Frequency Encoding

- Encodes the categories in a categorical feature by the frequency of their appearance in the dataset

A	0.44 (4 out of 9)
B	0.33 (3 out of 9)
C	0.22 (2 out of 9)

Feature	Encoded Feature
A	0.44
A	0.44
A	0.44
A	0.44
B	0.33
B	0.33
B	0.33
C	0.22
C	0.22

Feature Engineering of Temporal features - Time

- Apply binning on time data to make it categorial and more general.
- Binning a time in hours or periods of day, like below.

Hour range	Bin ID	Bin Description
[5, 8)	1	Early Morning
[8, 11)	2	Morning
[11, 14)	3	Midday
[14, 19)	4	Afternoon
[19, 22)	5	Evening
[22-24) and (00-05]	6	Night

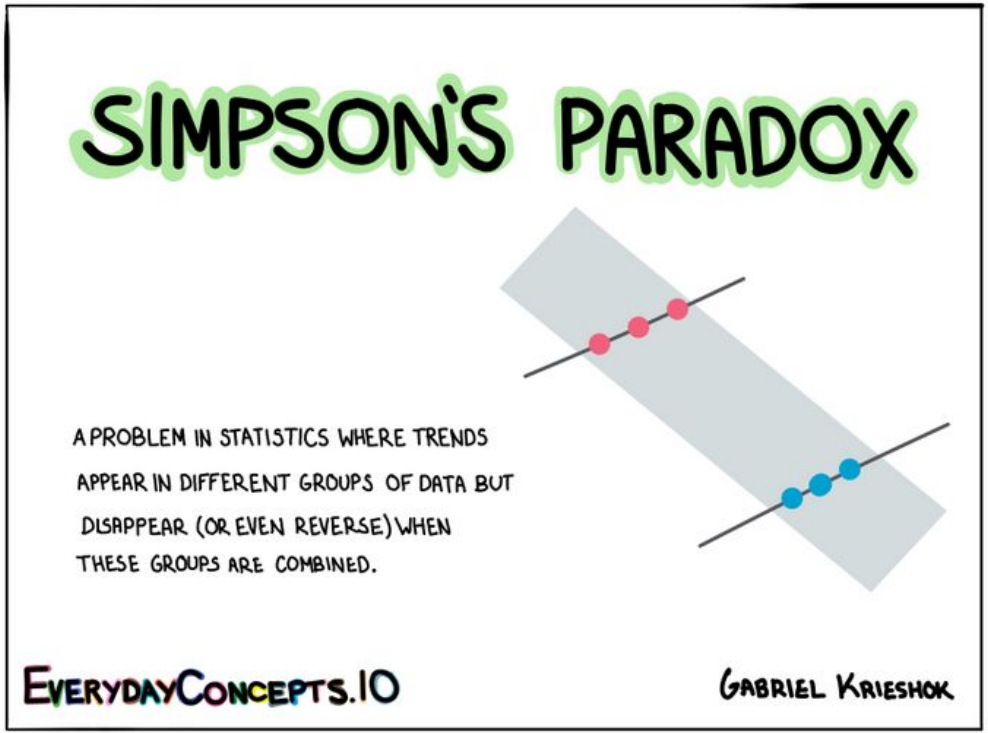
Feature Engineering of Temporal features - Trends

- Instead of encoding total spend, encode things like: Spend in last week, spend in last month, spend in last year.

- Gives a trend to the algorithm: two customers with equal spend, can have wildly different behavior — one customer may be starting to spend more, while the other is starting to decline spending.

- **Holiday Trends:** Simpson's paradox?

Feature Engineering of Temporal features - Simpson's paradox



Feature Engineering of Spatial features - Location

- Spatial variables encode a location in space, like:
  - GPS-coordinates (lat. / long.) - sometimes require projection to a different coordinate system
  - Street Addresses - requires geocoding
- Derived features
  - Distance between a user location and searched landmarks
  - Fraud detection : Using travel speed for detection

Breakout Ideation!

5 min
(3-4 per room)

- **What can be feature engineering techniques for textual data?**
- Share **3-4 ideas** and designate *one person to share*

Feature Engineering of Textual Data

Stats

Number of characters

Number of words

Number of capital characters

Number of punctuations

Number of words in quotes

Number of sentences

Number of unique words

Number of stop words

Average word Length

Average sentence Length

unique words vs words count

stop words vs words count

Feature Engineering of Textual Data

Cleaning

Lowercasing ➜ normalizing

Convert accented characters

Removing non-alphanumeric

Removing stop words

Tokenizing

Encode punctuation marks

Tokenize

N-Grams

Skip-grams

Char-grams

Affixes

Feature Engineering of Textual Data

Removing

Stopwords

Rare words

Common words

Roots

Spelling correction

Chop

Stem

Lemmatize

Enrich

Entity Insertion / Extraction

Parse Trees

Reading Level

Feature Engineering of Textual features - Vectorization

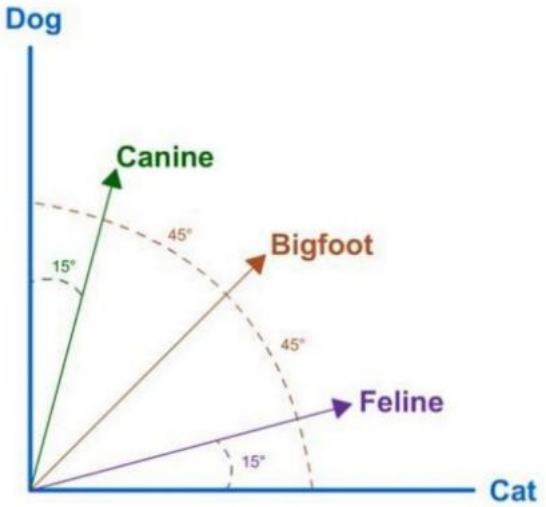
Represent each document as a feature vector in the vector space, where each position represents a word (token) and the contained value is its relevance in the document.

- BoW (Bag of words)
  - consider the following sentences:
    "The cat sat on the mat", "The dog chased the cat"
  - first, we create a vocabulary of all the unique words:
    ["The", "cat", "sat", "on", "the", "mat", "dog", "chased"]
  - Then we create numerical vectors for each sentence:
    [1, 1, 1, 1, 1, 1, 0, 0] and [1, 1, 0, 0, 1, 0, 1, 1] for the example sentences

- TF-IDF (Term Frequency - Inverse Document Frequency)
- Embeddings (eg. Word2Vec, Glove)
- Topic models (e.g LDA)

Feature Engineering of Textual features – Cosine Similarity

- Cosine Similarity is a measurement that quantifies the similarity between two or more vectors.
- The cosine similarity is the cosine of the angle between vectors.
- The vectors are typically non-zero and are within an inner product space.



$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}},$$

Data vs Algorithms

“More data beats clever algorithms, but better data beats more data.”

– Peter Norvig

Importance of Features

"...some machine learning projects succeed and some fail. Where is the difference? Easily the most important factor is the features used."

– Pedro Domingos

Feature Selection

Difference between Feature Extraction and Selection

Feature selection chooses a subset of features

Feature extraction creates new features (dimensions)

Reasons for doing Feature Selection

- It enables the machine learning algorithm to train faster
- It reduces the complexity of a model and makes it easier to interpret
- It improves the accuracy of a model if the right subset is chosen
- It reduces overfitting

Feature Selection Strategies

Here are three common feature selection strategies:

- Filter Methods
- Wrapper Methods
- Embedded Methods

Filter Methods

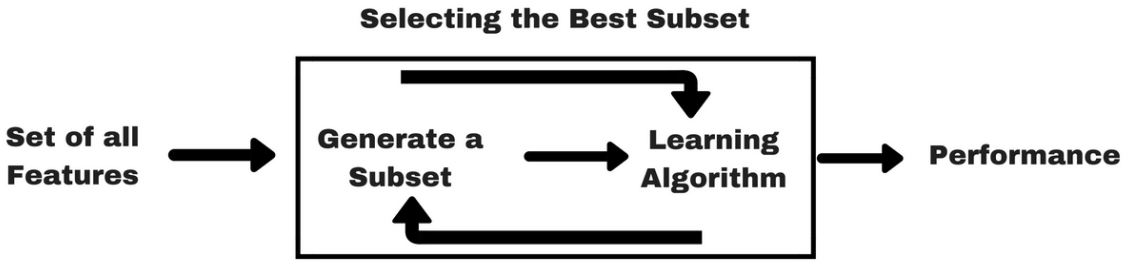
- Filter methods are generally used as a preprocessing step.
- The selection of features is independent of any machine learning algorithms.
- Instead, features are selected based on their scores in various statistical tests for their correlation with the outcome variable.

Wrapper Methods

- In wrapper methods, we try to use a subset of features and train a model using them.

- Based on the inferences that we draw from the previous model, we decide to add or remove features from your subset.

- The problem is essentially reduced to a search problem, and these methods are usually computationally very expensive.



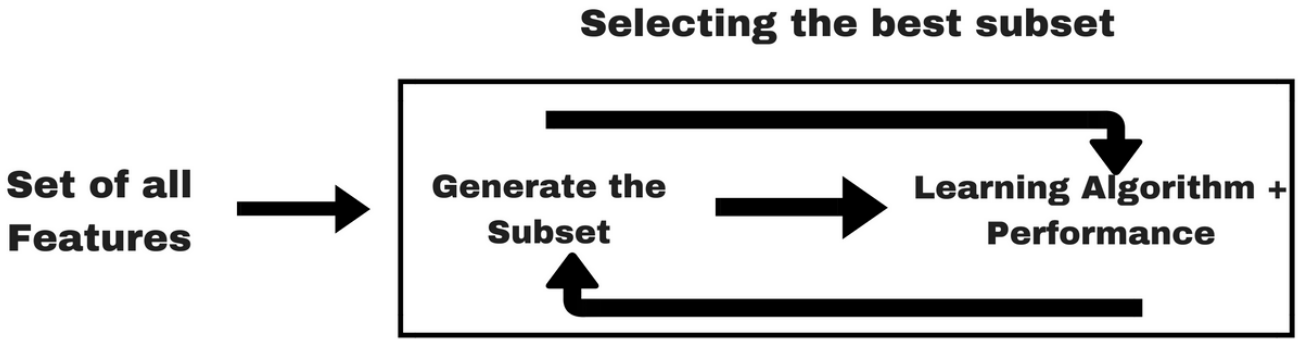
Review: Filter vs Wrapper Methods

Filter Method	Wrapper Method
Measures the relevance of features by their correlation with dependent variable	Measures the usefulness of a subset of feature by actually training a model on it
Much faster	Slower (involves model training)
Uses statistical methods for evaluation	Uses cross-validation
May fail to provide the best subset of features	Generally provides the best subset of features
Makes the model less prone to overfitting	Makes the model prone to overfitting

Embedded Methods

- Embedded methods combine the qualities of filter and wrapper methods. It's implemented by algorithms that have their own built-in feature selection methods.

- Some of the most popular examples of these methods are LASSO and RIDGE regression which have inbuilt penalization functions to reduce overfitting.



🤖 This Week!

Concepts

- Data Engineering Workflows
- Data Wrangling & Exploratory Data Analysis
- Feature Selection & Engineering
- **Data Leakage**
- Building Pipelines

Hands-On Activities

- Exploring and wrangling structured data for sales prediction pipeline
- Building Airflow workflow

Discussion

What do you think?

- What do you think Data Leakage means?

Data Leakage

- Data leakage is when information from outside the training dataset is used to create the model.

- In other words it is the problem of using information in your test samples for training your model. The problem with data leakage is that it inflates performance estimates.

Can you give some examples of Data Leakage?

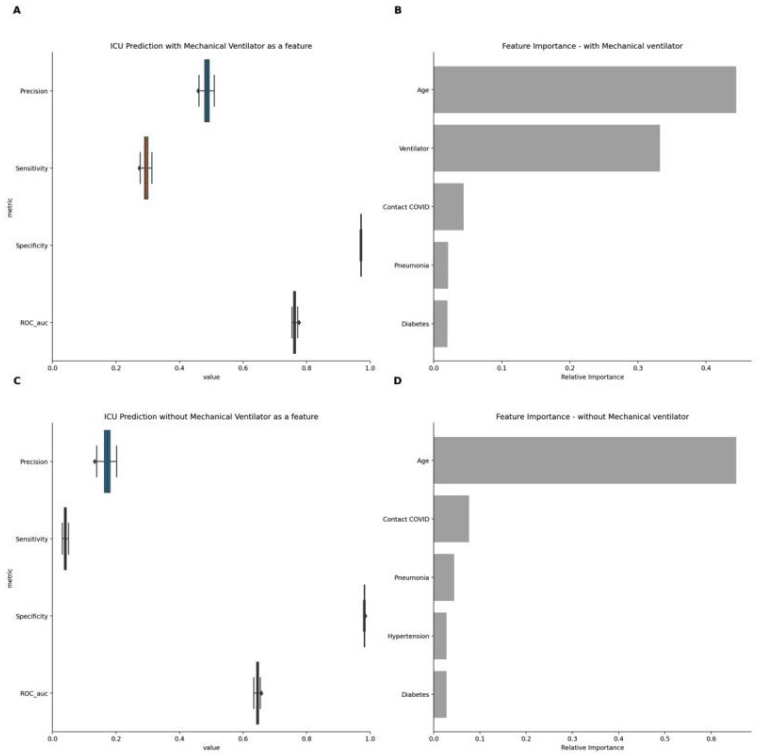
Link

Example of Data Leakage

(Fiho et al., 2021)

- Test the effect of including mechanical ventilation to predict intensive care unit (ICU) admission among patients with COVID-19

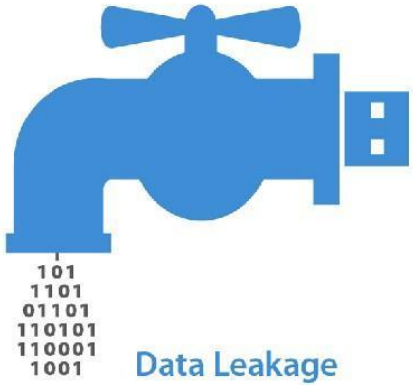
- Mechanical ventilation usually only occurs after ICU admission and should not be used to predict its risk



Situations where Data Leakage is a problem?

- Data Leakage may lead to reversing an anonymization, it can result in a privacy breach that you did not expect.

- It is a problem when you are developing your own predictive models. You may be creating overly optimistic models that are practically useless and cannot be used in production.



Data Leakage

The reality is that as a data scientist, you're at risk of producing a data leakage situation any time you prepare, clean your data, impute missing values, remove outliers, etc. You might be distorting the data in the process of preparing it to the point that you'll build a model that works well on your "clean" dataset, but will totally suck when applied in the real-world situation where you actually want to apply it.

— Page 313, Doing Data Science: Straight Talk from the Frontline

How to address Data Leakage?

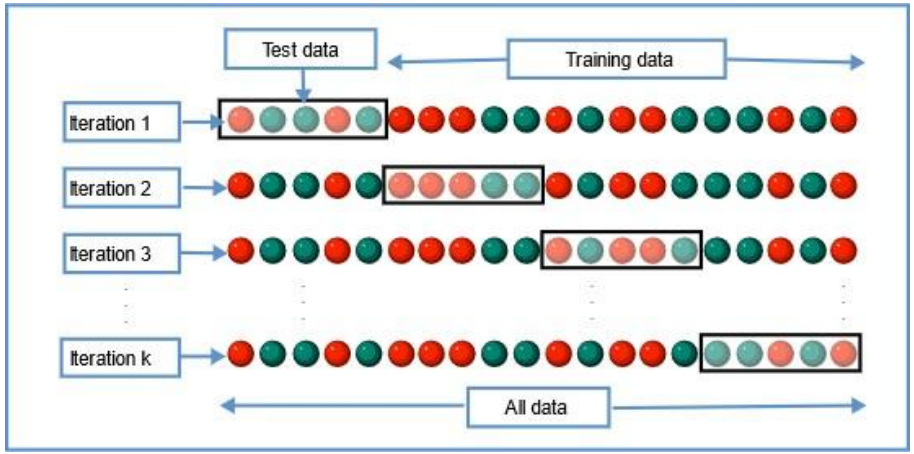
- **Create a Separate Validation Set:** Set aside a validation set in addition to training and test sets if possible. The purpose of the validation set is to mimic the real-life scenario and can be used as a final step. Can be used to detect overfitting

- **Apply Data preprocessing Separately to both Train and Test subsets:** Generally, in neural networks normalization is applied to the overall data set, which influences the training set from the information of the test set and eventually it results in data leakage. Hence, apply any normalization technique separately to both training and test subsets.

How to address Data Leakage?

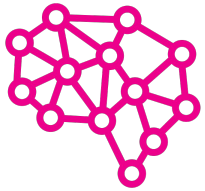
- **Cross-Validation Set:** Cross-validation splits the data into k folds and iterates over the entire dataset in k number of times and each time we are using k-1 fold for training and 1-fold for testing our model.

- The advantage of this approach is that we used the entire dataset for both training and testing purposes.



EDA Demo-House Prices - Advanced Regression Techniques (Kaggle dataset)

- Import data
- Data cleaning (removing duplicates, removing columns , missing values)
- Counts and stats and transformations
- Distributions
- Correlations and feature selections
- box plots, histograms
- Scatter plots and heatmaps

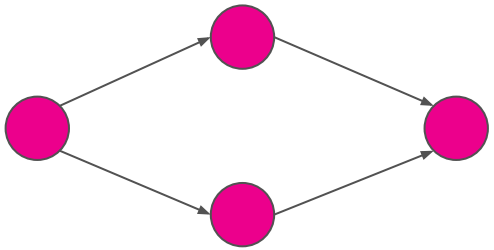


FourthBrain

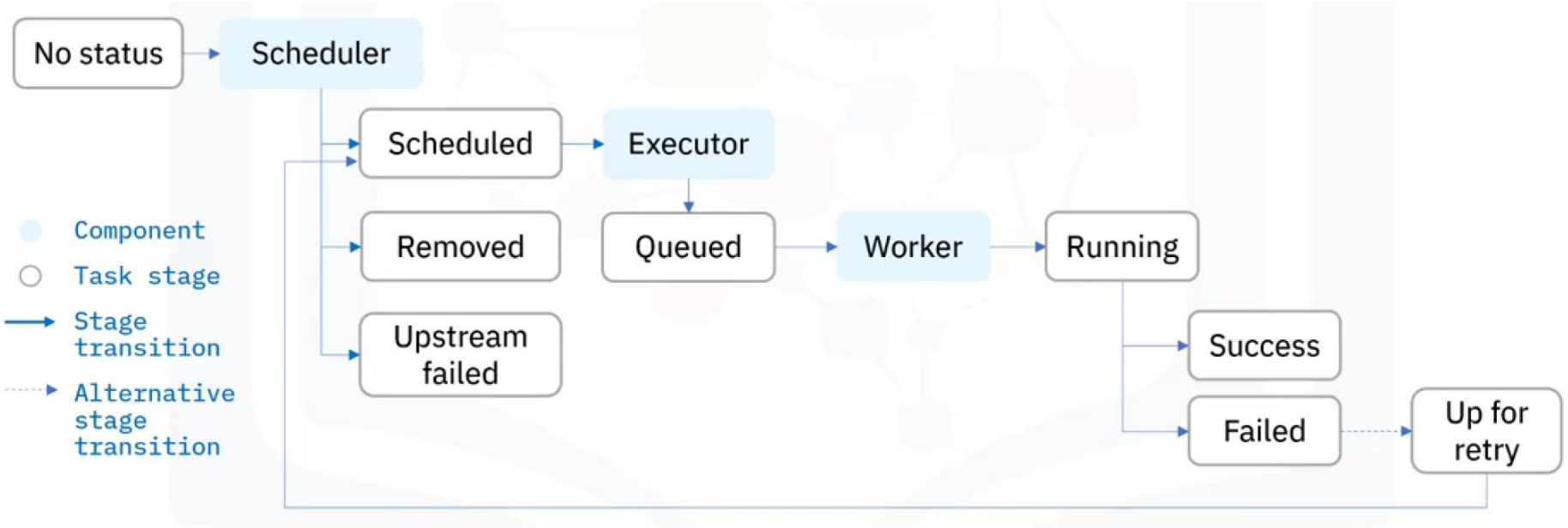
Airflow Overview

Airflow Overview

- Developed by Airbnb
- It is an open source tool allowing to programmatically author, schedule, and monitor workflows
- A workflow is represented as a DAG (Directed Acyclic Graph):
  - Graph consists of nodes and edges
  - Directed graph means that each edge has its direction
  - Acyclic means no loops



Airflow Task state lifecycle



Airflow Features And Benefits

- Pure Python
- User-friendly UI
- Plug and Play Integration with many services
- Easy to Use with Unlimited Pipeline Scope
- Open Source
- The workflows can be scheduled
- Built on 4 principles: Scalable, Dynamic, Extensible, Lean

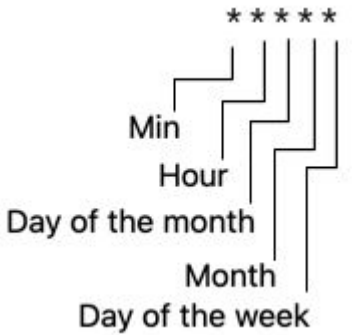
Airflow Use Cases

DAG definition components

```python
from datetime import datetime

from airflow import DAG
from airflow.decorators import task
from airflow.operators.bash import BashOperator

# A DAG represents a workflow, a collection of tasks
with DAG(dag_id="demo", start_date=datetime(2022, 1, 1), schedule="0 0 * * *") as dag:

    # Tasks are represented as operators
    hello = BashOperator(task_id="hello", bash_command="echo hello")

    @task()
    def airflow():
        print("airflow")

    # Set dependencies between tasks
    hello >> airflow()
```

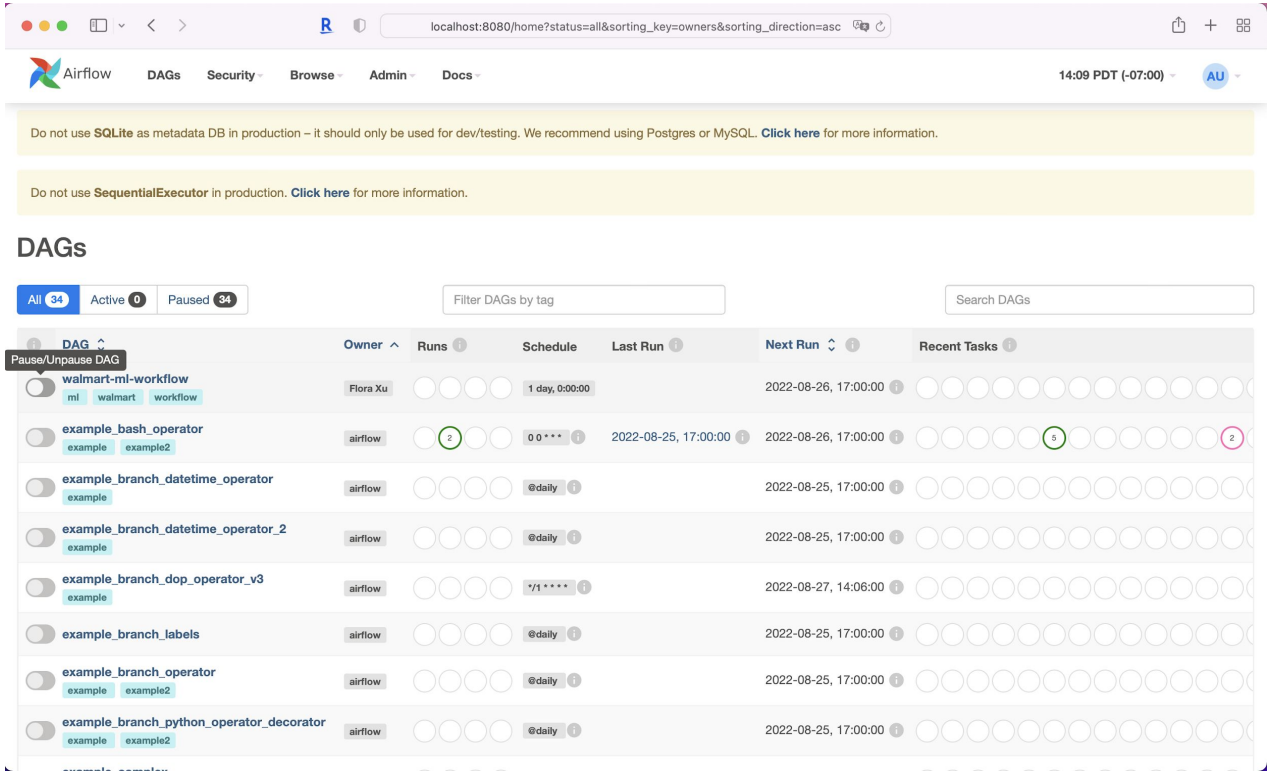
Job schedule format

A schedule is defined using the unix-cron string format (* * * * *) which is a set of five fields in a line, indicating when the job should be executed.



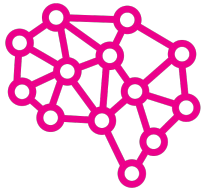
Sample schedule	Format
Every minute	* * * * *
Every Saturday at 23:45 (11:45 PM)	45 23 * * 6
Every Monday at 09:00 (9:00 AM)	0 9 * * 1
Every Sunday at 04:05 (4:05 AM)	5 4 * * SUN
Every weekday (Mon-Fri) at 22:00 (10:00 PM)	0 22 * * 1-5

Airflow User Interface



Airflow

Installation Instructions



FourthBrain

Coding Session

Pair Programming Practices

- higher-quality and faster delivery
- knowledge sharing
- building relationships
- How?
  - agree on how long you want each person to be the driver
  - one person shares their screen and after a certain amount of time you switch
  - each person should get multiple turns

Coding Assignment

- Due Friday, January 13th

- Expectations:
  - Read and run all the steps in the notebook
  - Finish up all the coding sections where you see #YOUR CODE HERE
  - Answer all the questions where you see YOUR ANSWER HERE
  - submit your notebook preferably by providing a link to your repo otherwise just submit your notebook
  - Common question - should I add everything to the current MLE11 repo - YES

EDA with Walmart Sales Data

- Tasks I-III
  - Load data
  - target, features and distributions
  - Impact from holidays
- Tasks IV-VI + Auto EDA
  - Visualize relationship between macroeconomic & external factors and sales
  - Feature engineering
  - Pipeline
  - Auto EDA
- Airflow

EDA with Walmart Sales Data

- **What is the assignment about?**
  - Linear Regression

$$y = X \cdot w + \varepsilon$$

  - SciKit Learn

https://scikit-learn.org/stable/user_guide.html

Feedback on Lecture and Concepts?

See you next week!