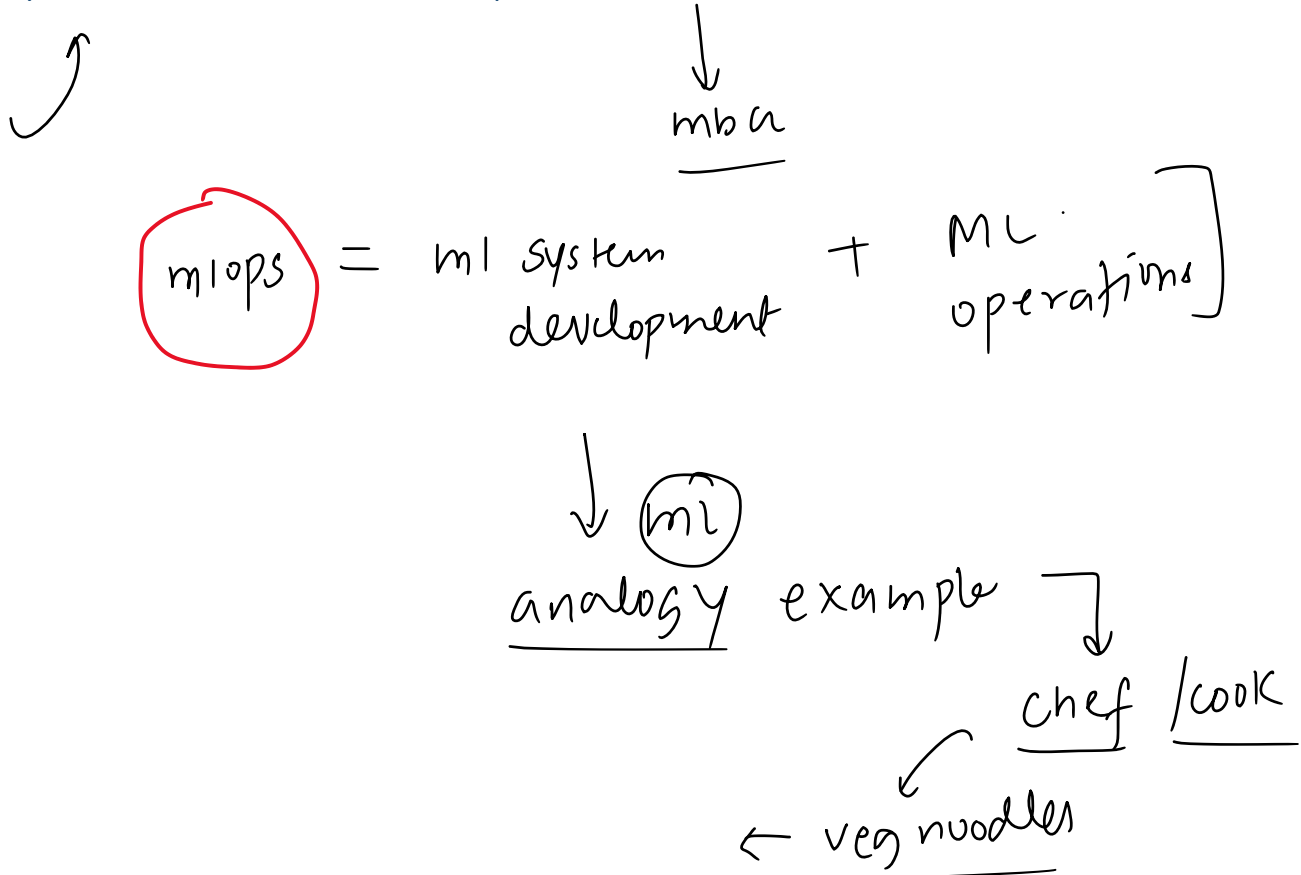


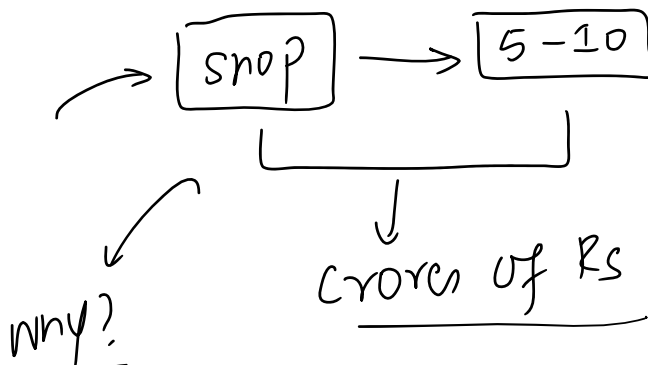
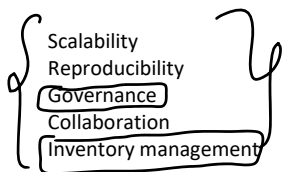
# What is MLOps

08 May 2024 13:50

MLOps, short for Machine Learning Operations, is a set of practices that aims to streamline and automate the lifecycle of machine learning models. This discipline merges machine learning (ML) system development and Machine Learning operations (Ops) to deliver consistent and efficient deployment and maintenance of ML models in production. The goal of MLOps is to bridge the gap between the development of ML models and their operational deployment, ensuring that they can be effectively scaled and maintained within an operational environment.



website → 1



why?

cross up :-

[5-10] noodle

10-100 → scale

+ 1 shop → more shops  
+ capital / human  
+ Reproducibility  
+ standardization

→ Automation

→ resource man  
+ inventory

→ Collaboration

→ Governance legal

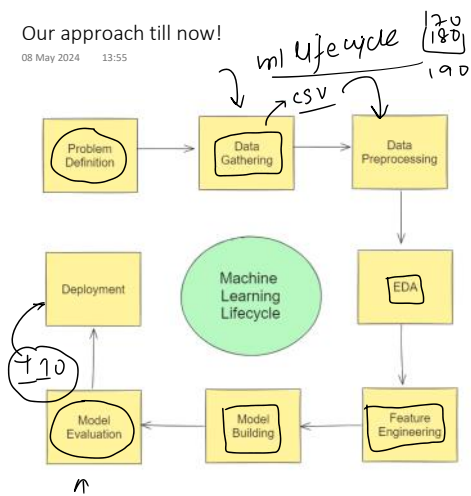
(ml)

product + operations

[model  
development]

# Our approach till now!

08 May 2024 13:55



ESPN Cricinfo

Live Scores Series Teams News Features Videos Stats IPL 2024

Delhi Capitals vs Rajasthan Royals

RR chose to field.

Current RR: 6.54

Live Scorecard Live Blog Commentary Live Stats Overs Playing XI Table Videos Photos News Fantasy

	R	B	4s	6s	SR	This Bowler	Last 5 Balls	Mat	Runs	HS	Ave
<b>BATTERS</b>											
Abhishek Patel* (nb)	5	4	1	0	125.00	5 (4b)	5 (4b)	26	501	73*	26.37
Jake Fraser-McGurk (nb)	7	7	1	0	100.00	1 (1b)	1 (1b)	44	911	84	23.97
<b>BOWLERS</b>											
Sandeep Sharma (rm)	0	0	0	0	7.20	2	1	0	0.5	0	6
Trent Boult (lrm)	1	0	6	0	6.00	4	1	0	1	0	6

Partnership: 12 Runs, 11 B (RR: 6.54)

Reviews Remaining: Delhi Capitals - 2 of 2, Rajasthan Royals - 2 of 2

56 Sanju Samson has broken the record for playing most IPL matches (56) as captain for RR, going past Shane Warne

Match centre Ground time: 19:40 Scores: M Venkat Raghavi | Comments: @himanshu\_a30

1.5 Sandeep to Mitchell Patel, no run

WIN PROBABILITY RR 54.04%

DC innings RR innings

Current Over 2 • DC 12/0

Live Forecast: DC 173

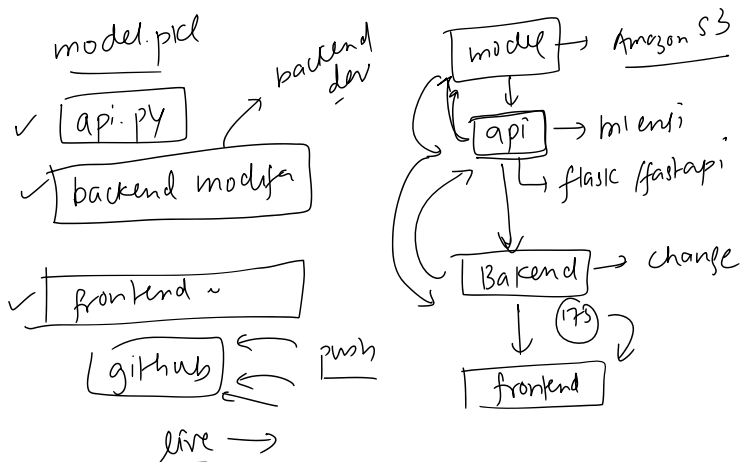
Powered by Smart Stats

MATCH COVERAGE All Match News

Live Blog - Delhi Capitals vs Rajasthan Royals - Hetmyer, Jurel out for Royals; Ishant back for Capitals

Delhi Capitals host Rajasthan Royals at Arun Jaitley stadium. Get all your live stats, analyses and colour right here on ESPNcricinfo's live blog.

Tristan Stubbs - superb against spin, destructive at the death



# Problem 1 - The Data Problem

09 May 2024 09:45

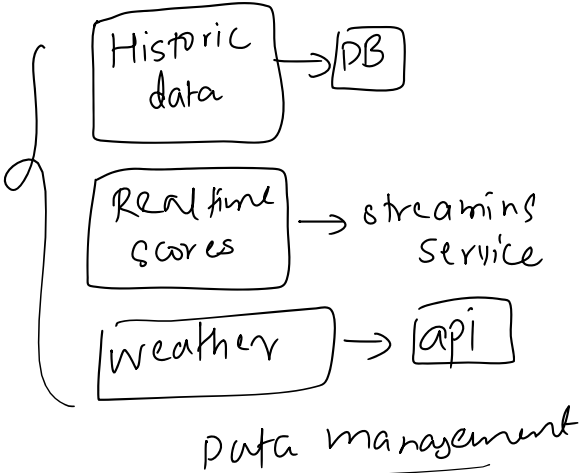
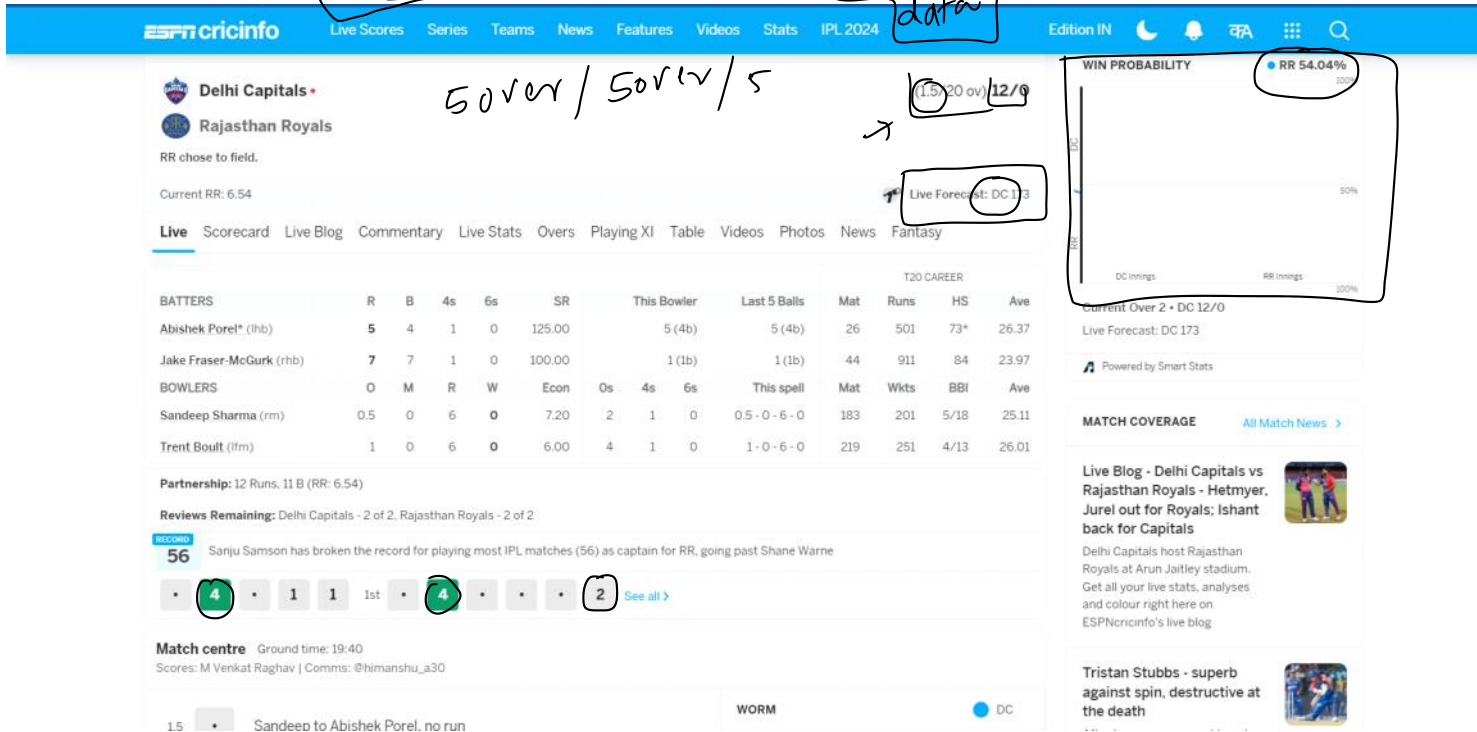
Weather

data

↳ historical

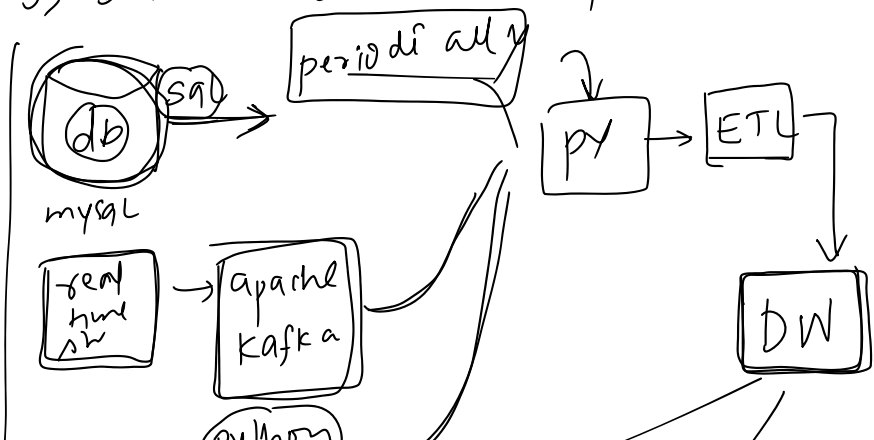
currently

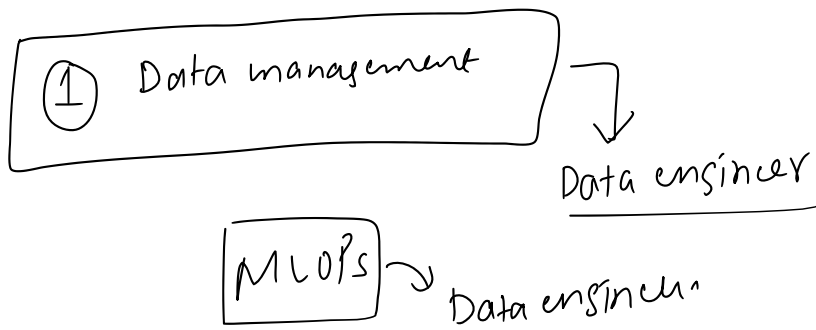
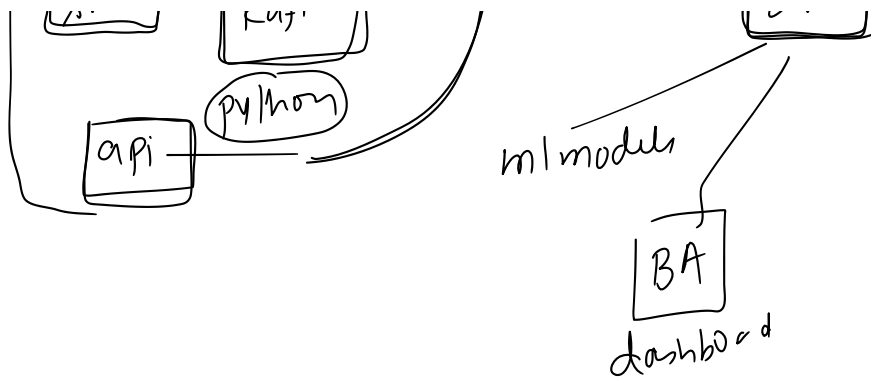
realtime data



CSV  
file  
PY  
read\_csv.

- 1) Data ingestion
- 2) Data transformation
- 3) Data storage 1 week





## Problem 2 - The code problem

09 May 2024 09:46

data management →

```
# Import necessary libraries
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score

# Load the Iris dataset
data = load_iris()
X = data.data
y = data.target

# Preprocess the data: Standardize features
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

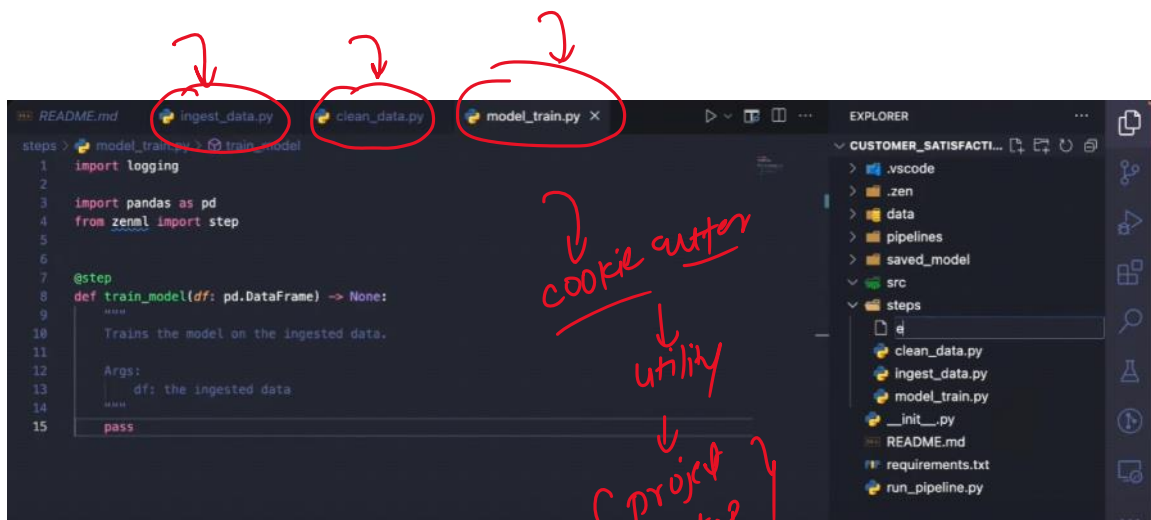
# Split the data into training and test sets
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.3, random_state=42)

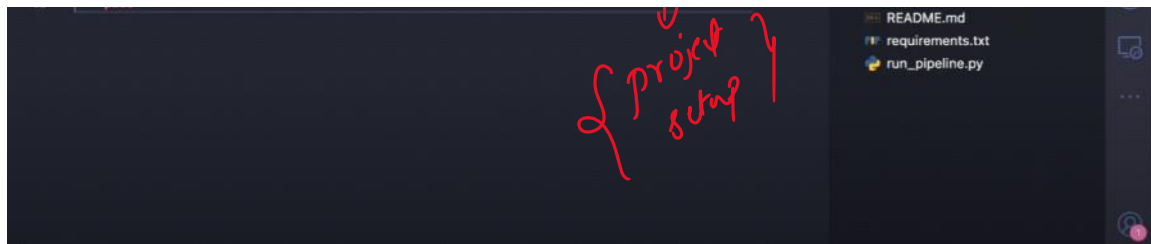
# Train a logistic regression model
model = LogisticRegression()
model.fit(X_train, y_train)

# Make predictions on the test set
y_pred = model.predict(X_test)

# Evaluate the model
accuracy = accuracy_score(y_test, y_pred)
print(f'Accuracy: {accuracy:.2f}')
```

1. **Difficulty in Maintenance**: As the complexity of the code increases, it becomes harder to maintain. Changes in one part of the code can unexpectedly affect other parts due to the tightly coupled nature of the implementation. This can lead to increased time spent debugging and verifying that changes do not break existing functionality.
2. **Limited Reusability**: In a non-modular setup, code reusability is minimal. Functions and components are often written to solve a specific problem and are tightly integrated with other parts of the code, making them difficult to extract and reuse in other projects or contexts.
3. **Harder Collaboration**: When code is not modular, it's more challenging for multiple developers to work on the same project simultaneously. Since everything is interconnected, developers need to be more cautious about the changes they make, which can slow down development and increase the risk of merge conflicts.
4. **Poor Scalability**: Scaling a non-modular application can be problematic. As more features and functionalities are added, the codebase can become unwieldy and difficult to manage. This lack of scalability extends not only to the size of the codebase but also to the handling of larger data sets or more complex models.
5. **Testing Challenges**: Testing a non-modular codebase is typically more challenging and less effective. Without clear boundaries between components, it's tough to perform unit testing; thus, most testing becomes integration testing, which is less granular and can overlook specific defects.
6. **Inefficiency in Iteration and Experimentation**: Machine learning projects often require iterative adjustments and experiments with different models, parameters, and data preprocessing methods. A non-modular approach complicates these experiments, as each change might require alterations across multiple sections of the code, increasing the risk of errors and the time needed for each iteration.
7. **Difficulty in Tracking Changes and Versioning**: In a non-modular architecture, it's challenging to implement effective version control practices. Changes to the codebase can affect many parts of the application, making it harder to track changes, roll back updates, or manage different versions of the code for different experiments or deployment scenarios.
8. **Integration Problems**: Integrating the system with other services or pipelines (like data ingestion, real-time data processing, etc.) can be cumbersome without clear module boundaries. This lack of separation can lead to issues when connecting different parts of a project or ensuring that they interact smoothly.





Good development  
practices

mlops

### Problem 3 - The versioning problem

09 May 2024

09:46

Score predictor → git / github

Versioning

collaborate → ↗  
↓  
versioning → rollback

software  
↓  
code

ml project  
code + data + model  
↓  
DVC →

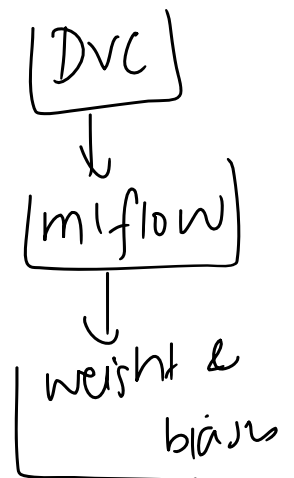
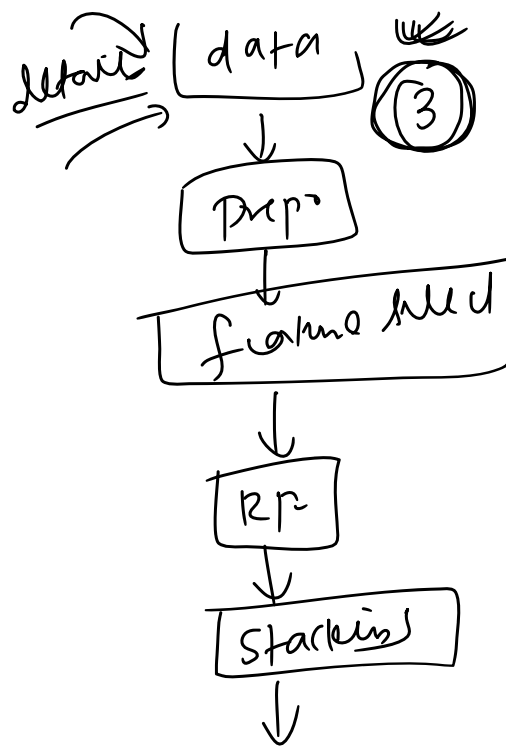
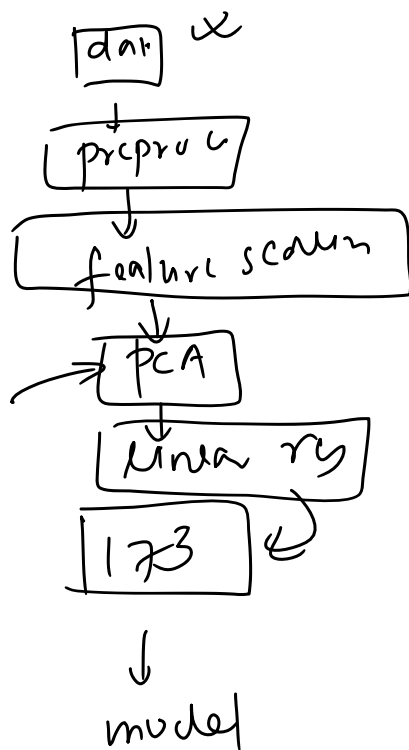


## Problem 4 - The Experimentation Problem

09 May 2024 09:47

Experiment Tracking

10 exp



# Problem 5 - Automation

09 May 2024 21:54

# Problem 6 - The [Deployment Problem]

09 May 2024 09:47

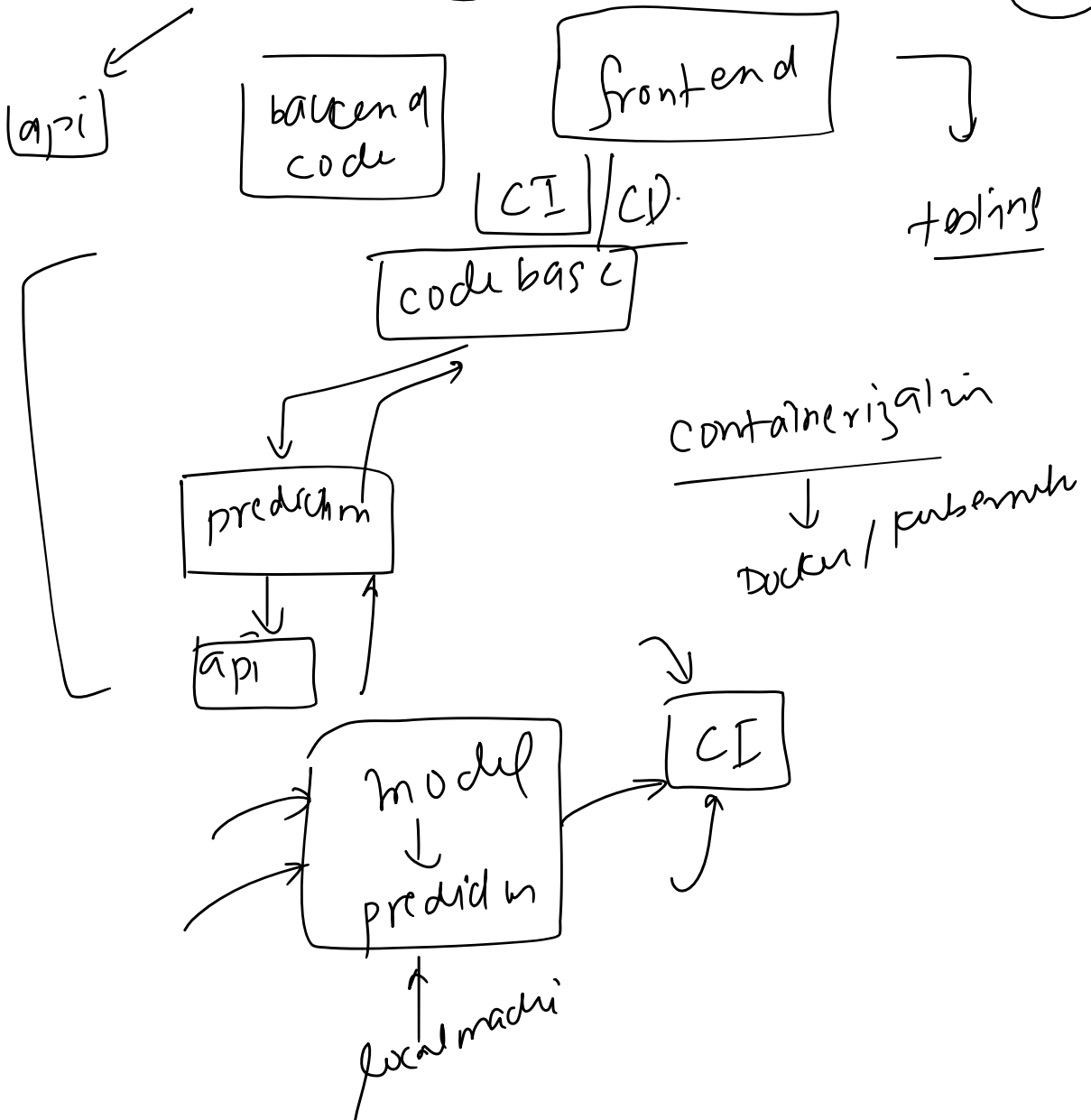
mlops

automate

DevOps

ml

model.pkl → (S3)



## Problem 7 - The Drift problem

09 May 2024 09:47

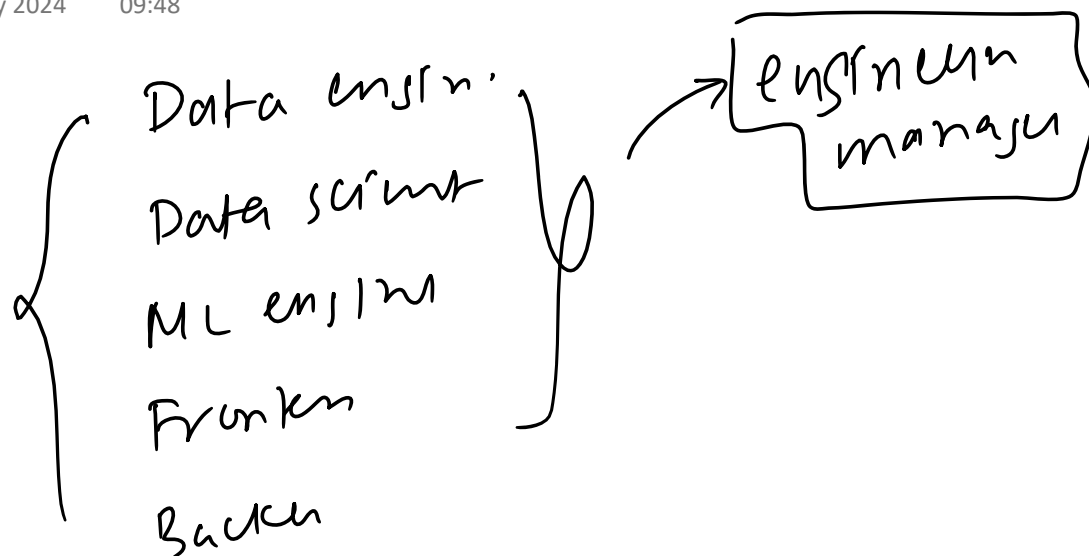
model deploy      monitoring  
grafana/prometheus  
Retraining

# Problem 8 - The Infrastructure Problem

09 May 2024 09:48

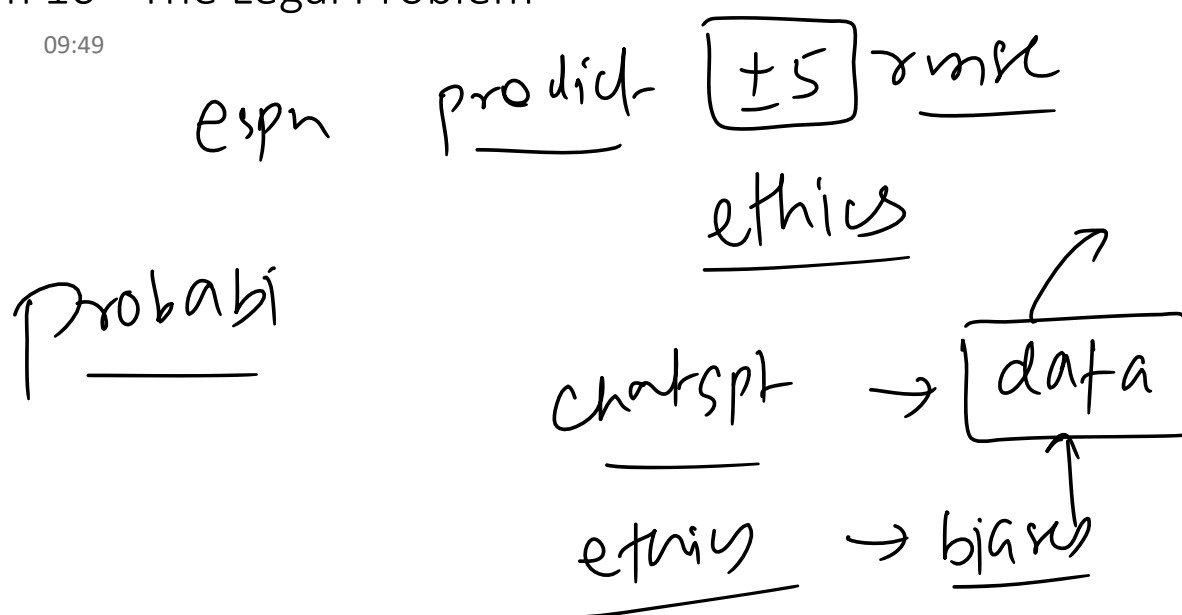
## Problem 9 - The Collaboration Problem

09 May 2024 09:48



## Problem 10 - The Legal Problem

09 May 2024 09:49



# Aspects of MLOps

08 May 2024 14:09

## 1. Data Management

ml dev

ml dev / ml ops

- a. Data Collection
- b. Data Preprocessing
- c. Data Validation
- d. Data Security
- e. Data Compliance
- f. Feature Store

## 2. Development Practices

dev

- a. Modular Coding

## 3. Version Control

dev

- a. Code versioning
- b. Data versioning
- c. Model versioning

## 4. Experiment Tracking

dev

- a. Tracking ml experiments
- b. Test and validation
- c. Model registry

## 5. Model Serving and CI/CD

ops

- a. Continuous Integration
- b. Containerization
- c. Continuous Deployment

## 6. Automation

ops

- a. Pipeline automation [Data ingestion pipeline, model training pipeline, model validation and testing, model deployment, model monitoring and retraining]
- b. Orchestration

→ High Level Knowledge  
↓  
PA 9

## 7. Monitoring and Retraining

ops

- a. Model Monitoring
- b. Drift Detection
- c. Retraining

## 8. Infrastructure Management

ops

- a. Cloud based solutions to handle scalability concerns
- b. Cost management
- c. Managing multiple vendors

## 9. Collaboration and Operations

ops



c. Managing multiple vendors

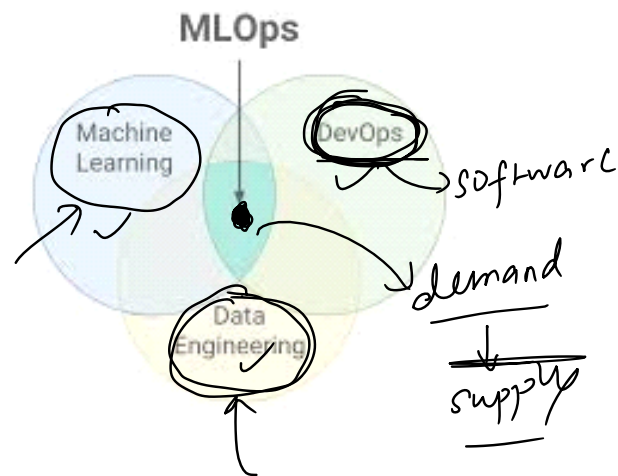
## 9. Collaboration and Operations ops

a. Unified workspace

b. Role based access

## 10. Governance and Ethics ops

**MLOps refers to the practice and discipline within machine learning that aims to unify and streamline the machine learning system development (Dev) and machine learning system operation (Ops). It involves collaboration between data scientists, ML engineers, and IT professionals to automate and optimize the end-to-end lifecycle of machine learning applications.**



# MLOps Maturity Levels

08 May 2024 14:19

# Benefit of MLOps

08 May 2024 14:19

1. Scalability
2. Improved performance
3. Reproducibility
4. Collaboration and efficiency
5. Risk reduction
6. Cost Savings
7. Faster time to market
8. Better compliance and governance

1. Data Management
2. Development Practices
3. Version Control
4. Experiment Tracking/Model Registry
5. Model Serving and CI/CD
6. Automation
7. Monitoring and Retraining
8. Infrastructure Management
9. Collaboration and Operations
10. Governance and Ethics

# Challenges

08 May 2024 14:19

1. Complexity of ml models [variability, black box nature]
2. Quality of data
3. Cost and resource constraints
4. Handling scale
5. Security risks
6. Compliance and regulatory concerns
7. Integration with existing systems
8. Limited Expertise/Skill gap

# Prerequisites to become a MLOps Engineer

08 May 2024 14:20

1. Basic understanding of ML
  - a. Cleaning and preprocessing
  - b. Feature engineering
  - c. Model building
2. Software development skills
  - a. Python
  - b. Git
  - c. Software development best practices [OOP, Design Patterns]
3. Data Engineering
  - a. SQL
  - b. Big Data Tech [Spark, Kafka]
  - c. Data Storage Solutions [Databases, Data Warehouses, Data lakes]
4. DevOps Principles and Tools
  - a. CI/CD Pipeline
  - b. Automation
5. Familiarity with cloud platforms
  - a. AWS, GCP and Azure
6. Containerization technologies
  - a. Docker
  - b. Kubernetes
7. Networking Principles
  - a. Distributed computing
8. Security Fundamentals
  - a. Cybersecurity fundamentals
9. Soft Skills