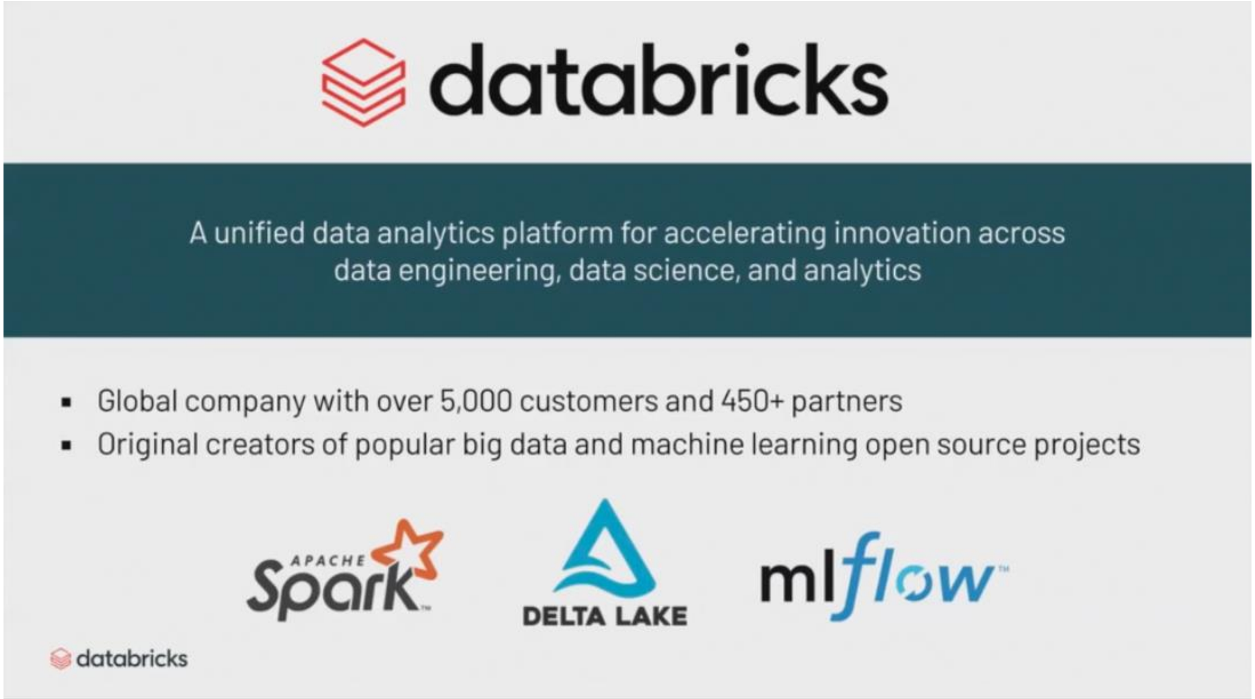


databricks

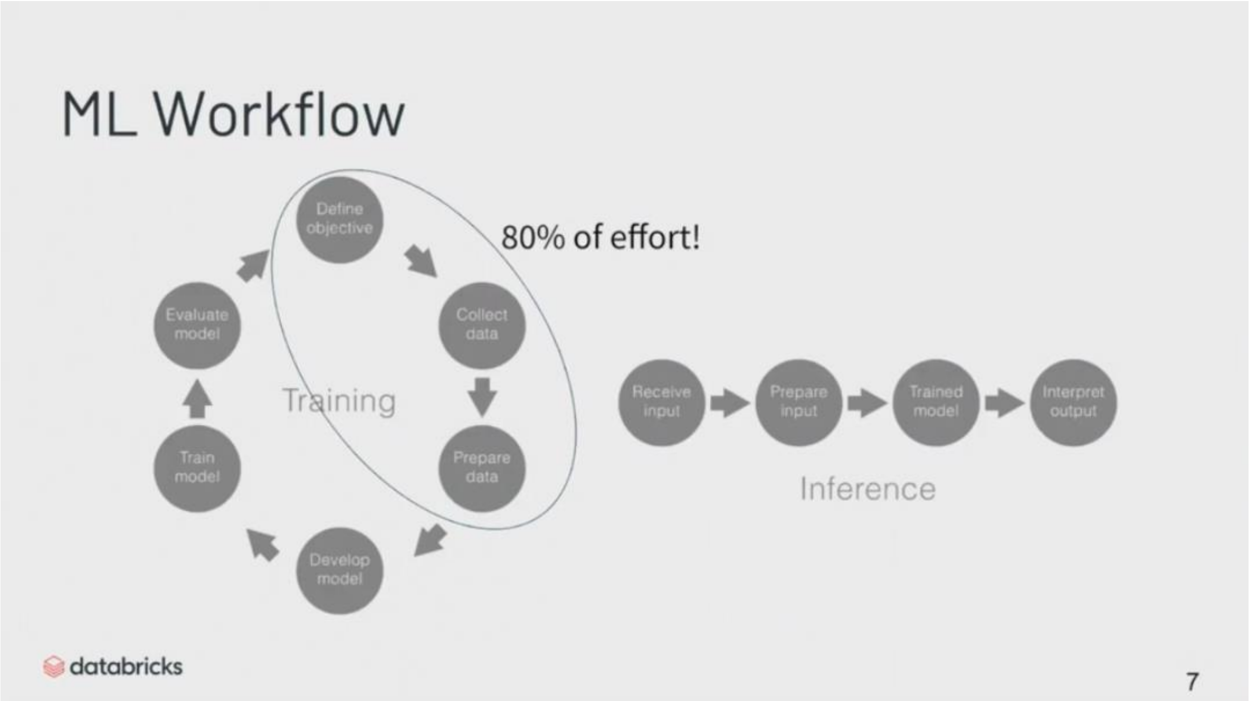
A unified data analytics platform for accelerating innovation across data engineering, data science, and analytics

- Global company with over 5,000 customers and 450+ partners
- Original creators of popular big data and machine learning open source projects

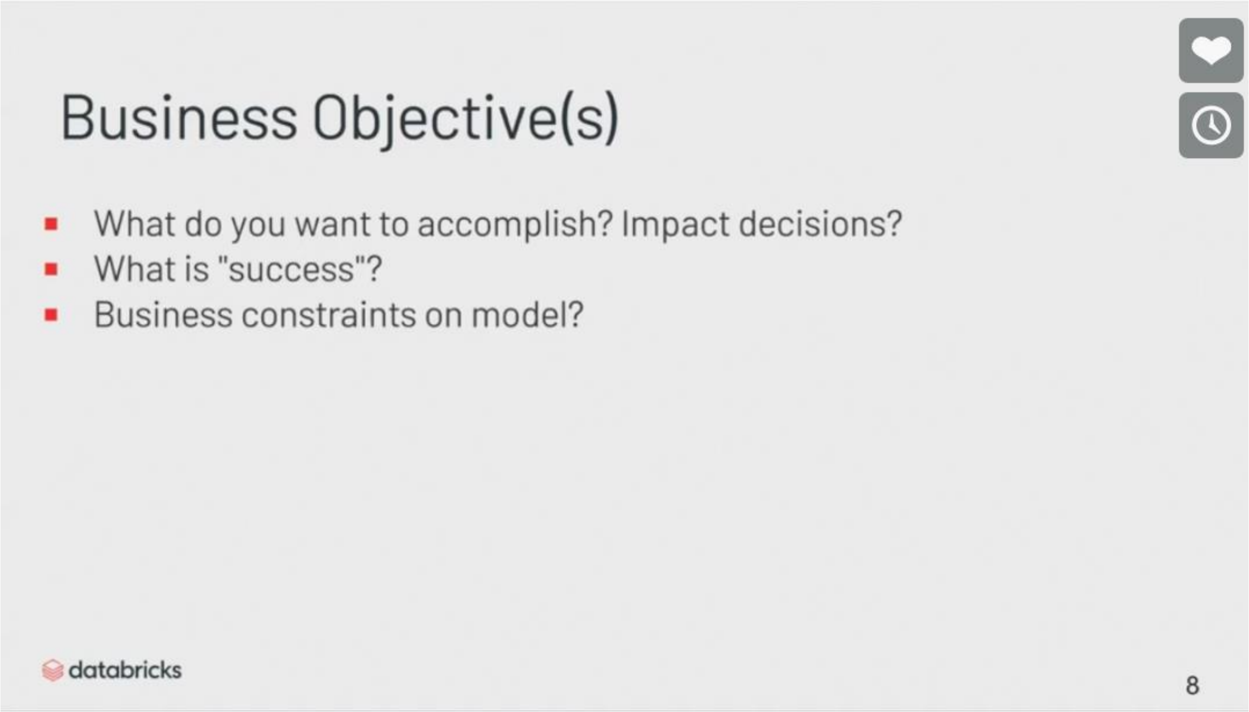
APACHE Spark
DELTA LAKE
mlflow
databricks

Setting up for success

databricks

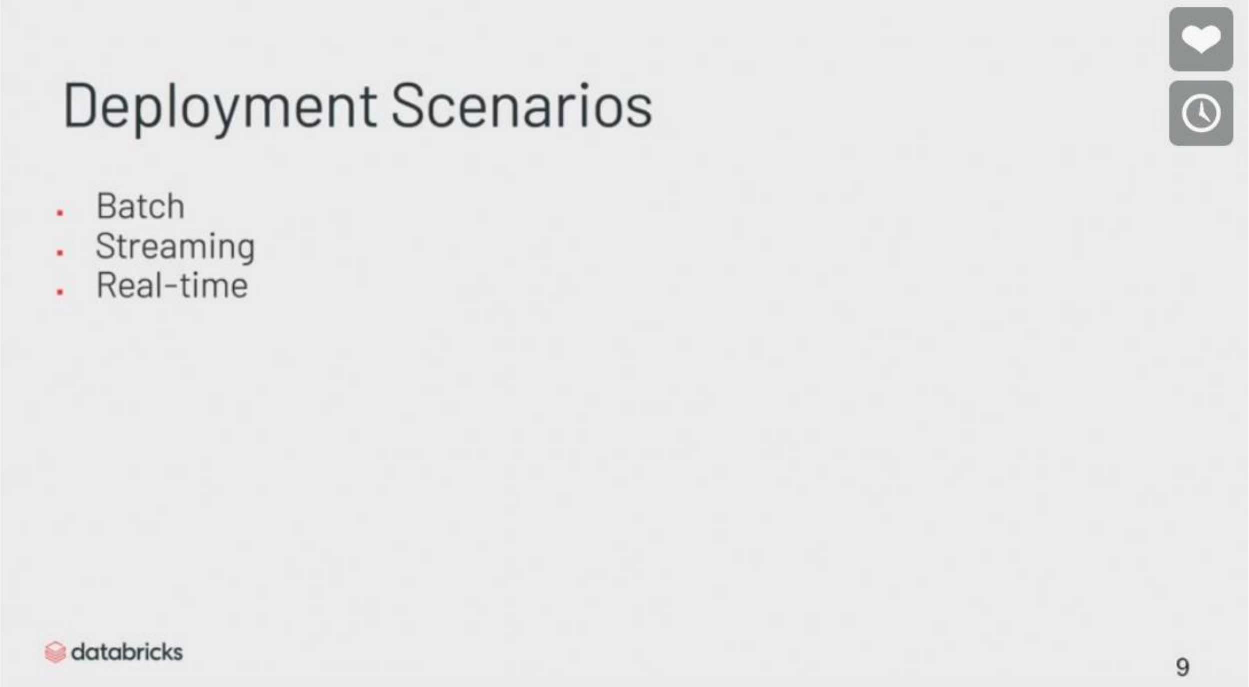


ML Workflow



Business Objective(s)

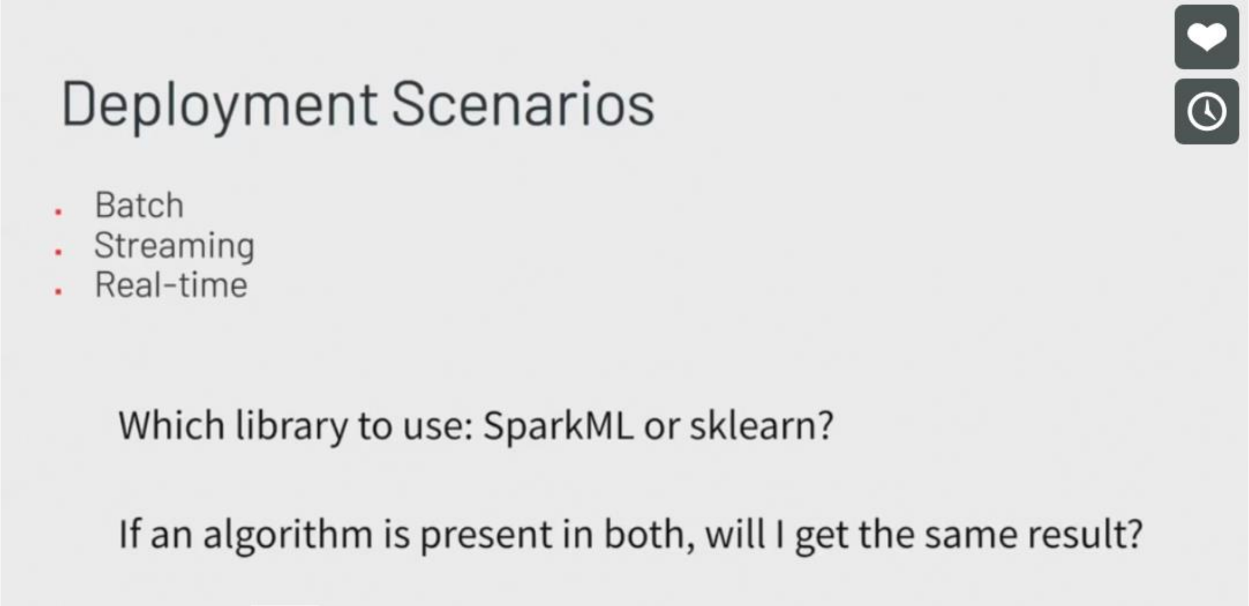
- What do you want to accomplish? Impact decisions?
- What is "success"?
- Business constraints on model?



Deployment Scenarios

- Batch
- Streaming
- Real-time

databricks

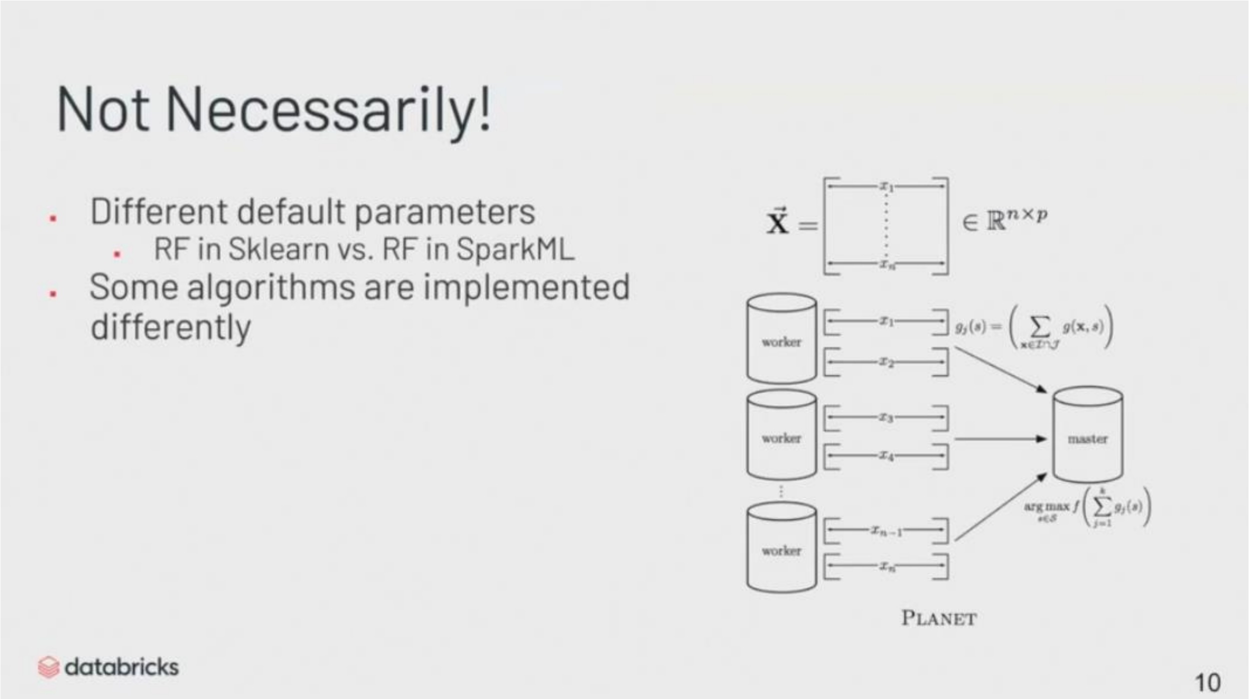


Deployment Scenarios

- Batch
- Streaming
- Real-time

Which library to use: SparkML or sklearn?

If an algorithm is present in both, will I get the same result?

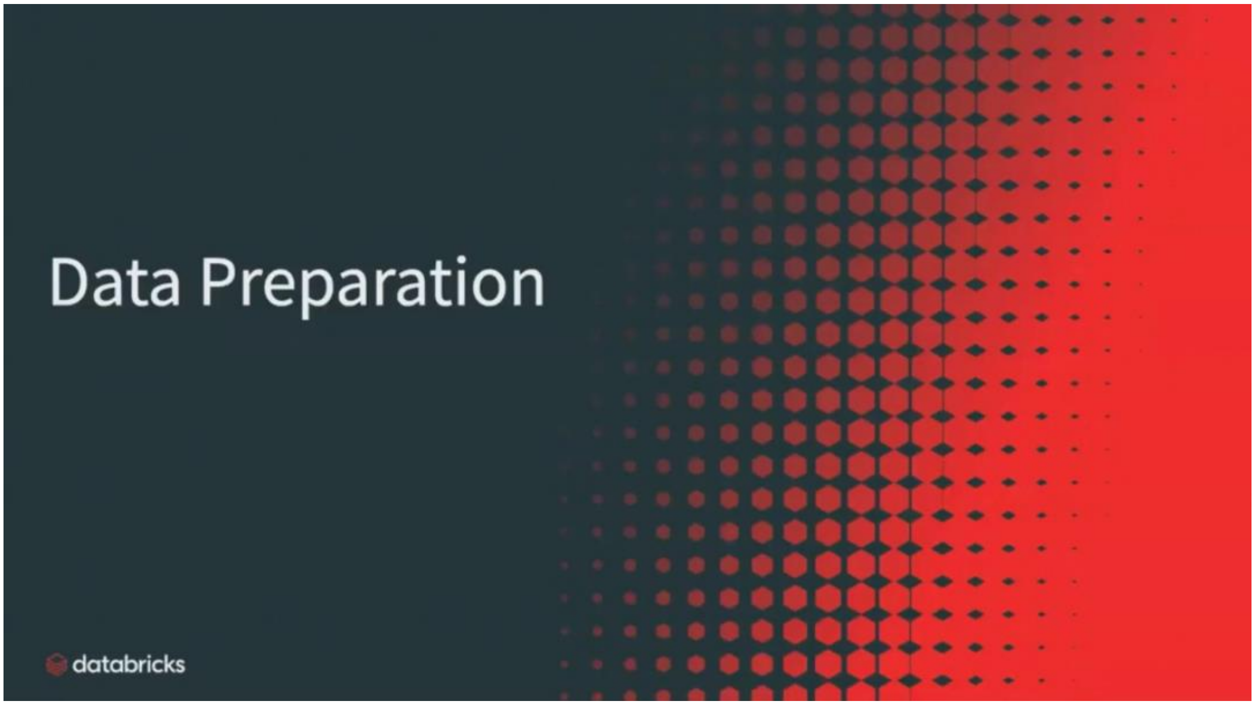


Not Necessarily!

- Different default parameters
  - RF in Sklearn vs. RF in SparkML
- Some algorithms are implemented differently

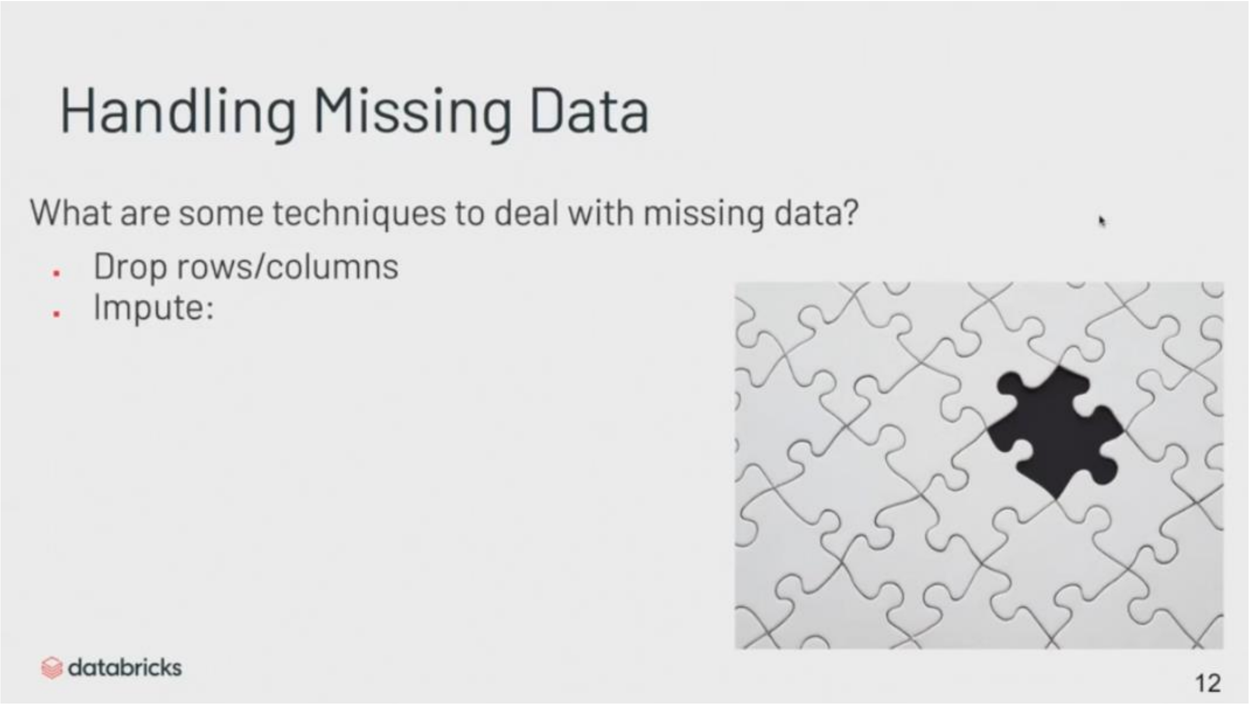
$\vec{\mathbf{X}} = \in \mathbb{R}^{n \times p}$
x_1
x_n
worker
worker
worker
x_1
x_2
x_3
x_4
x_{n-1}
x_n
$g_j(s) = \left(\sum_{\mathbf{x} \in \mathcal{I} \cap \mathcal{J}} g(\mathbf{x}, s) \right)$
master
$\arg\max_{s \in \mathcal{S}} f\left(\sum_{j=1}^{k} g_j(s) \right)$
PLANET

databricks



Data Preparation

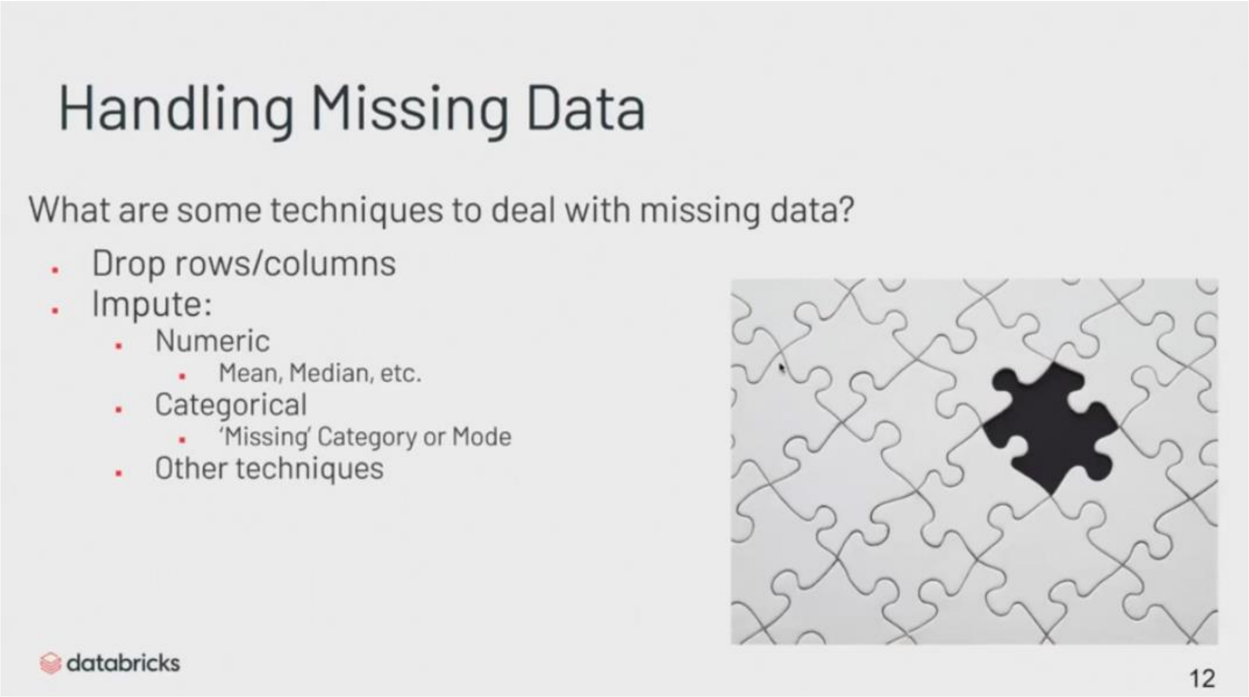
databricks



Handling Missing Data

What are some techniques to deal with missing data?

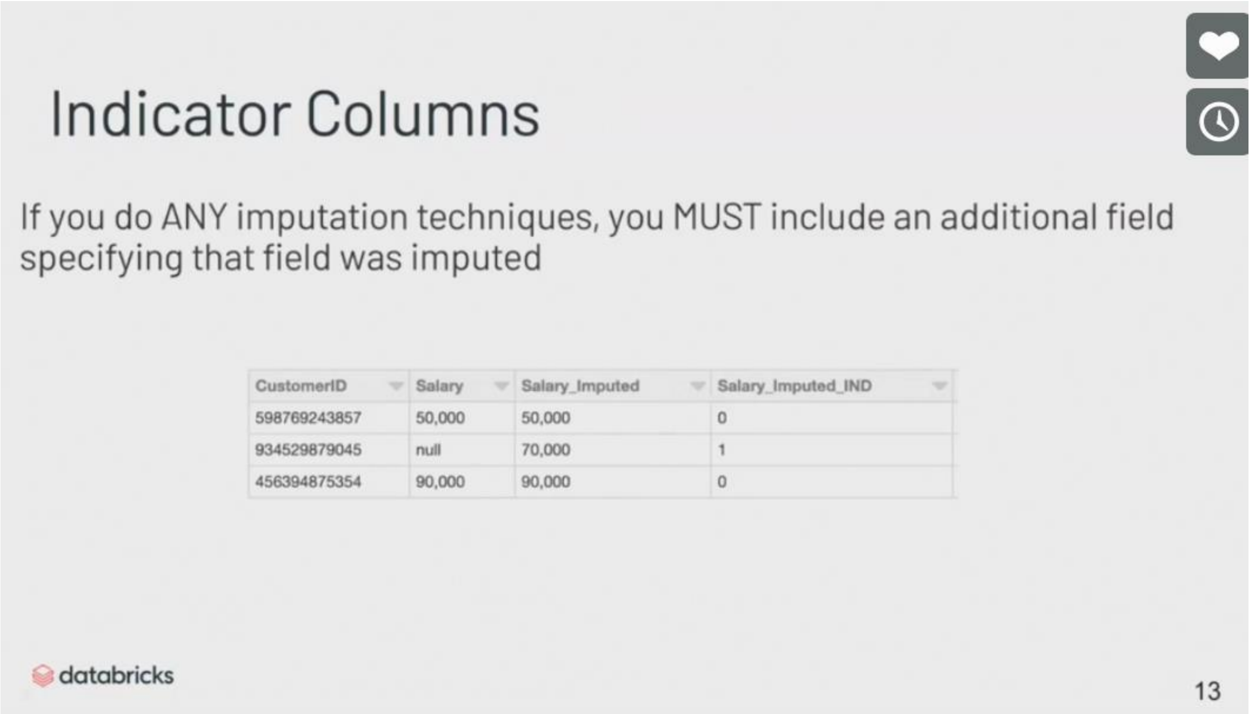
- Drop rows/columns
- Impute:



Handling Missing Data

What are some techniques to deal with missing data?

- Drop rows/columns
- Impute:
  - Numeric
    - Mean, Median, etc.
  - Categorical
    - 'Missing' Category or Mode
  - Other techniques

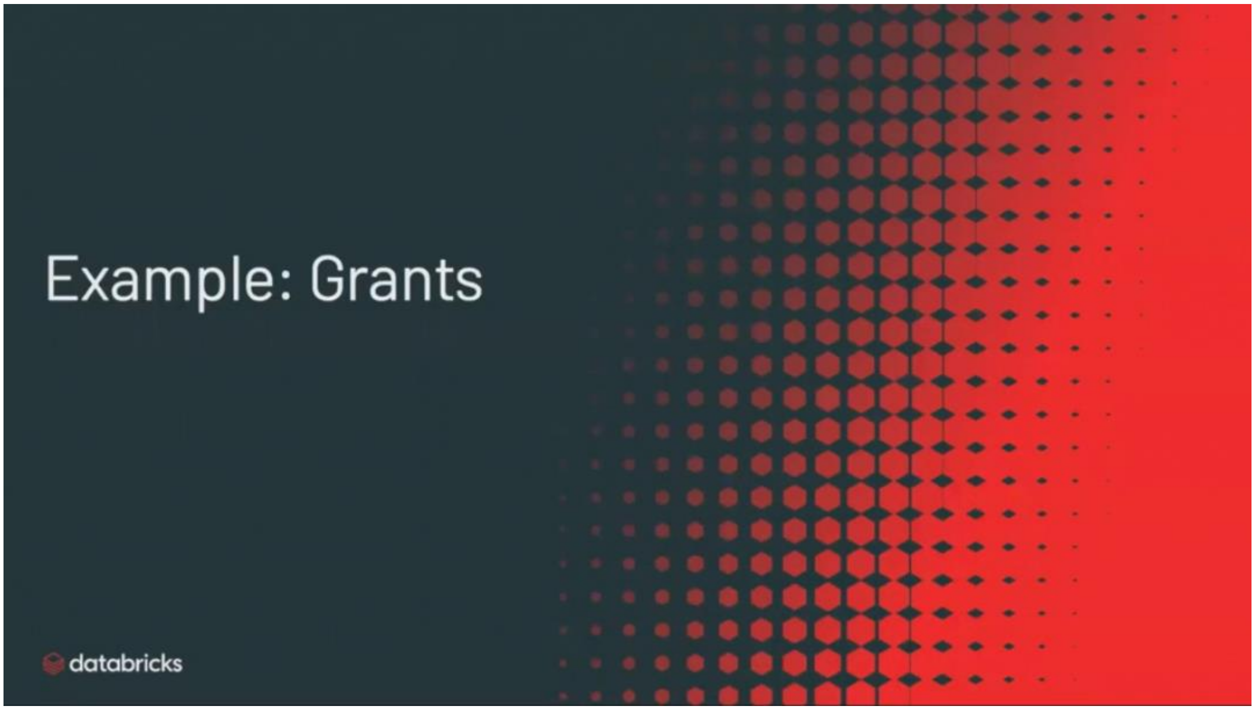


Indicator Columns

If you do ANY imputation techniques, you MUST include an additional field specifying that field was imputed

CustomerID	Salary	Salary_Imputed	Salary_Imputed_IND
598769243857	50,000	50,000	0
934529879045	null	70,000	1
456394875354	90,000	90,000	0

databricks



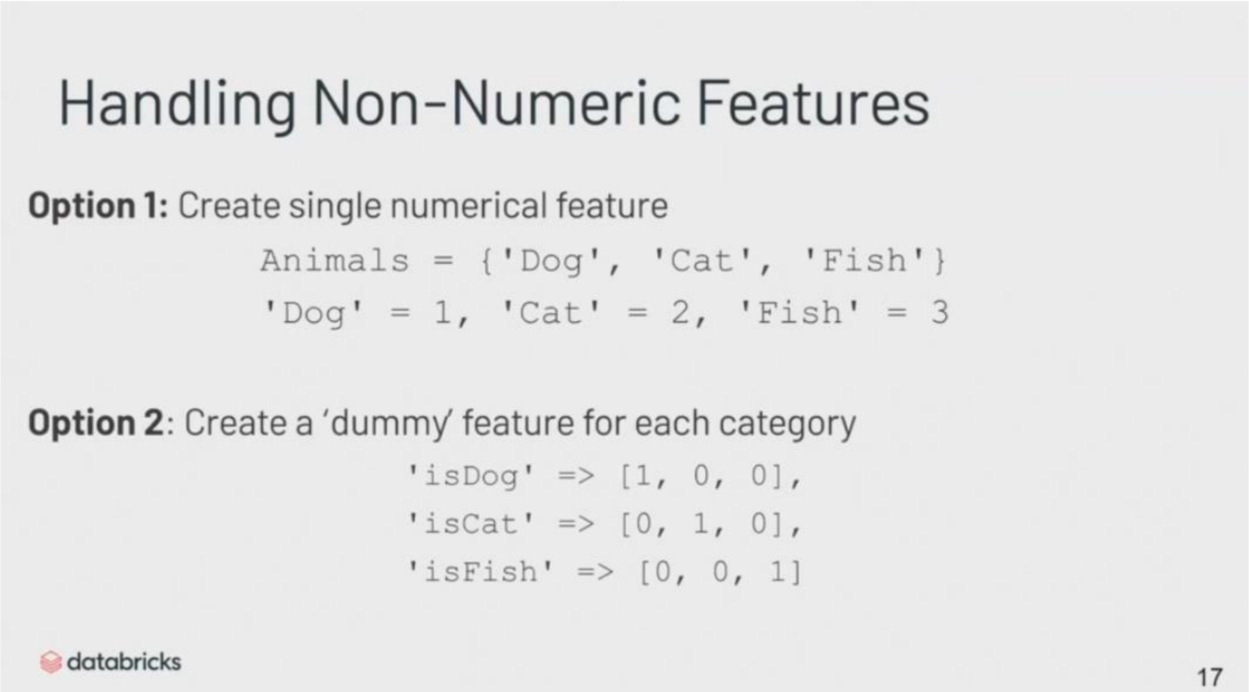
Example: Grants

databricks

Feature Engineering + Model Limitations

Feature Preparation

- Feature Engineering & modelling process are closely related.
- No “one size fits all” solution.



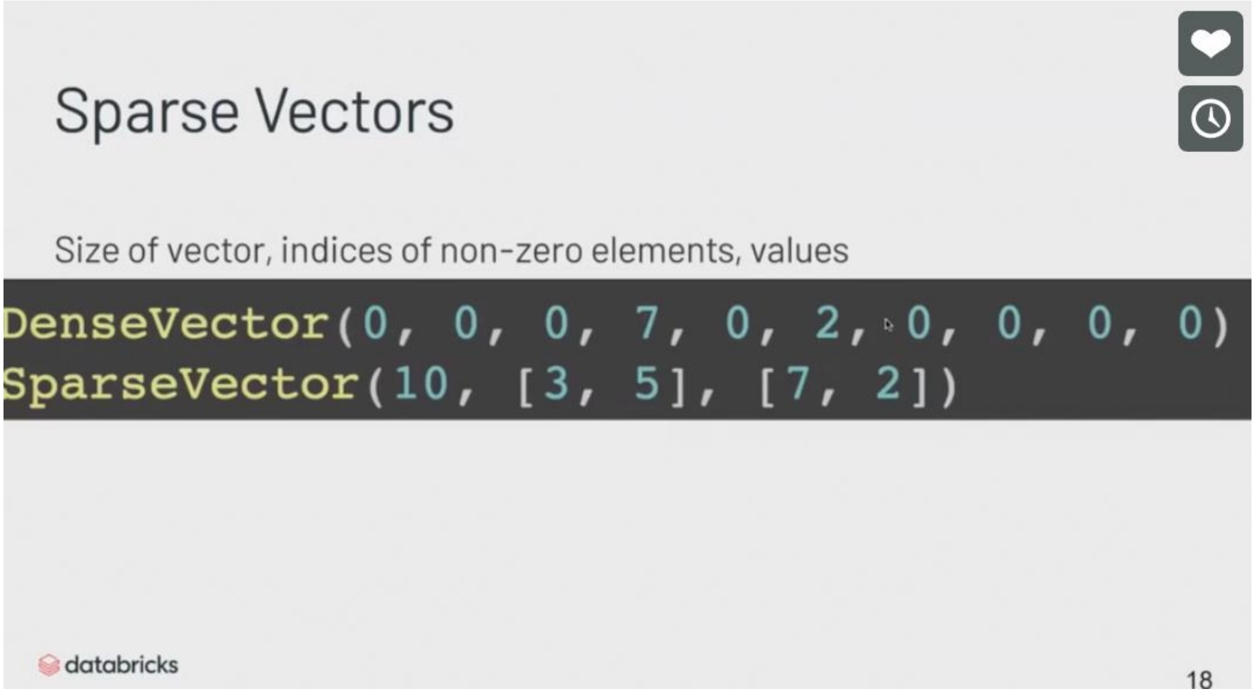
Handling Non-Numeric Features

Option 1: Create single numerical feature

```
Animals = {'Dog', 'Cat', 'Fish'}
'Dog' = 1, 'Cat' = 2, 'Fish' = 3
```

Option 2: Create a 'dummy' feature for each category

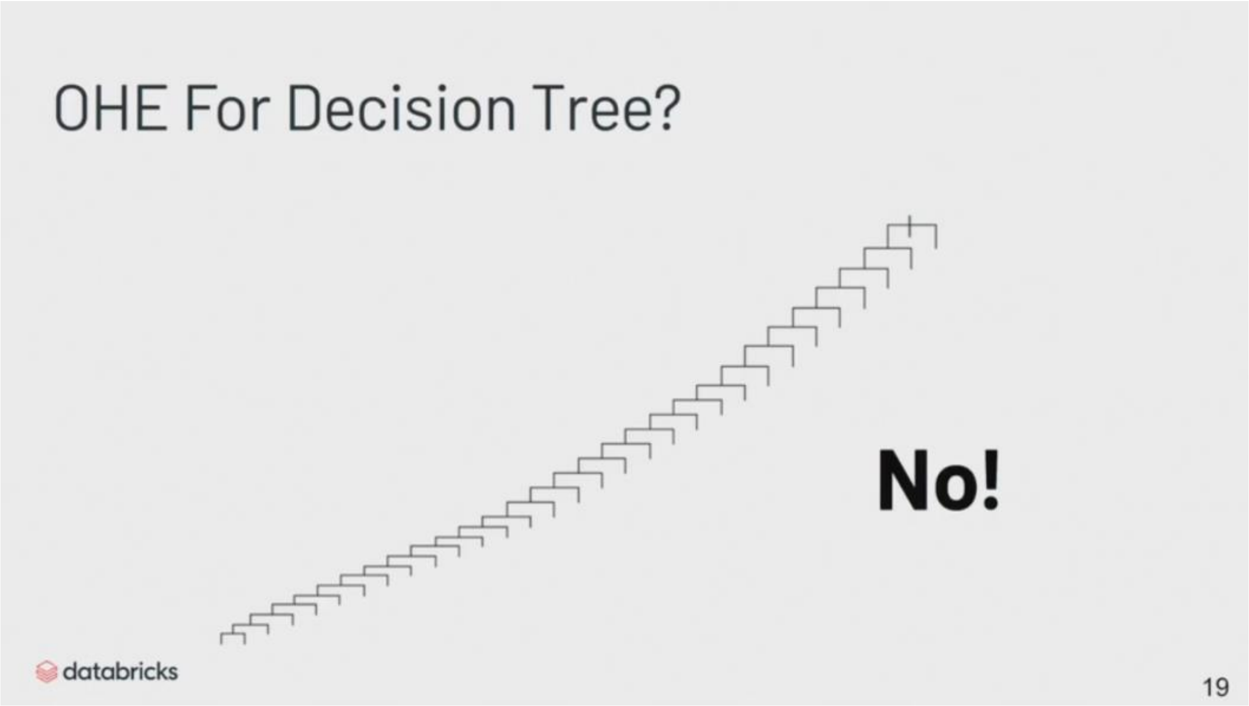
```
'isDog' => [1, 0, 0],
'isCat' => [0, 1, 0],
'isFish' => [0, 0, 1]
```



Sparse Vectors

Size of vector, indices of non-zero elements, values

```
DenseVector(0, 0, 0, 7, 0, 2, 0, 0, 0, 0)
SparseVector(10, [3, 5], [7, 2])
```



OHE For Decision Tree?

No!

databricks

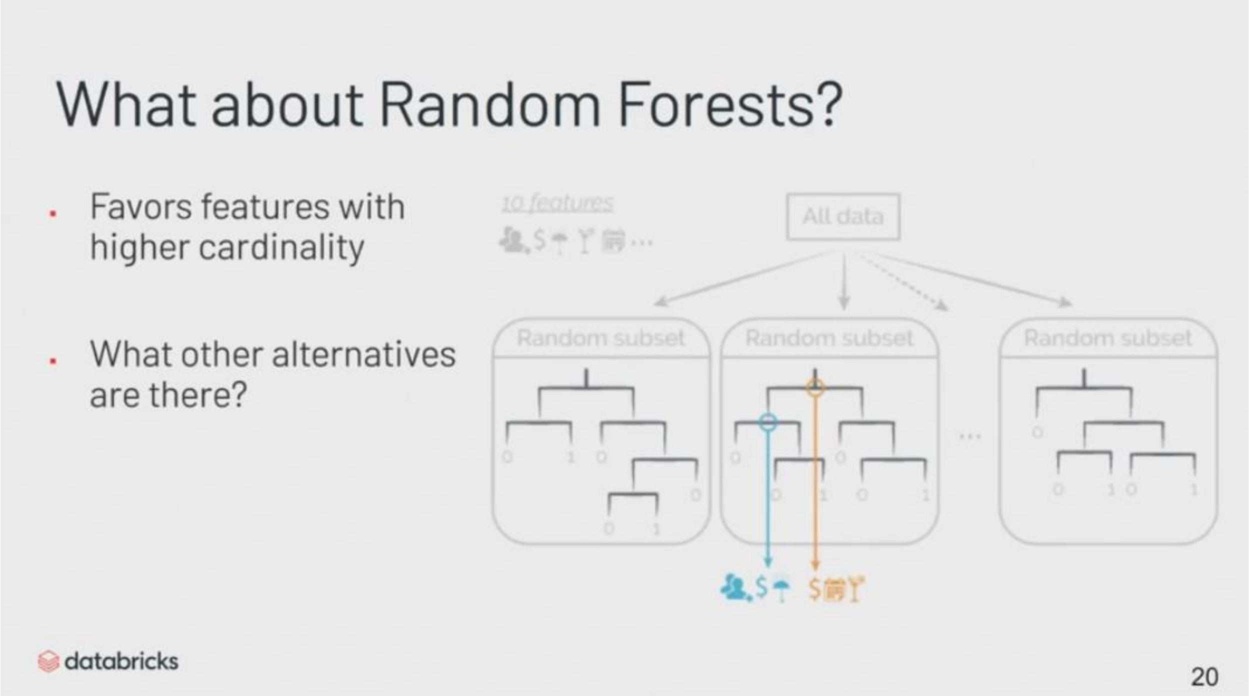
19

OHE good for linear algorithms

OHE bad for decision trees

Sklearn use ordinalEncoder or labelEncoder

Spark use stringIndexer()

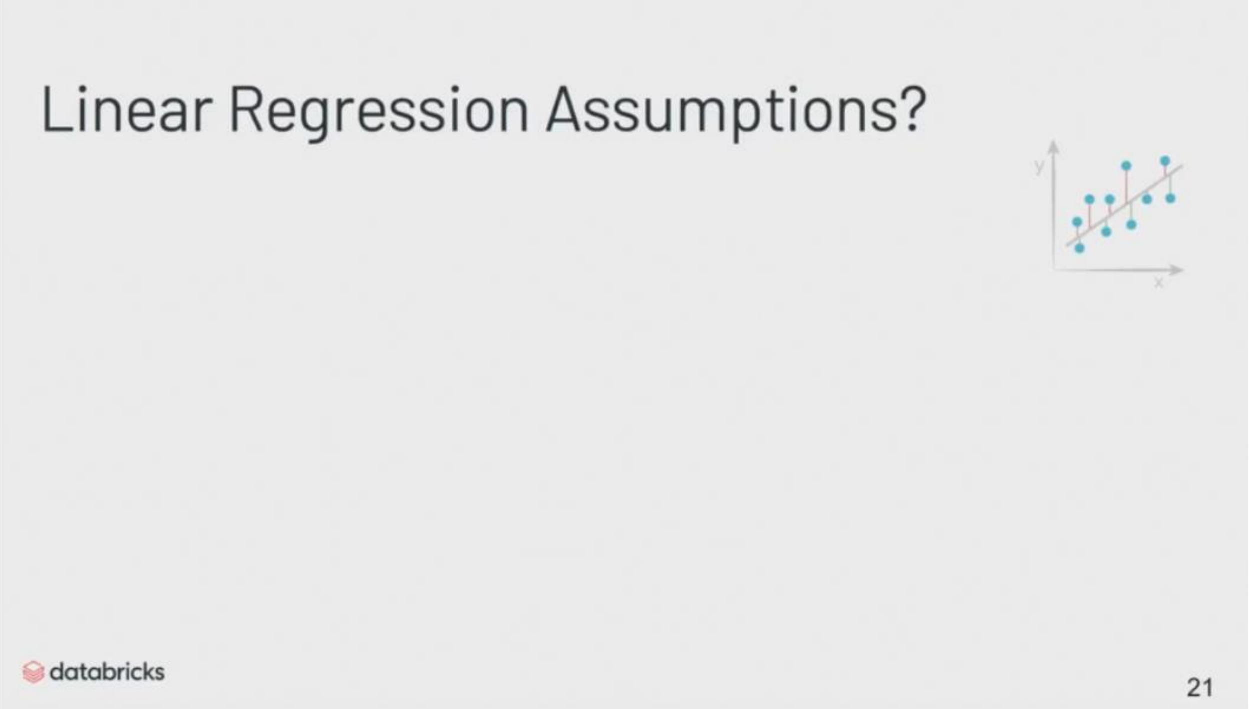


What about Random Forests?

- Favors features with higher cardinality
- What other alternatives are there?

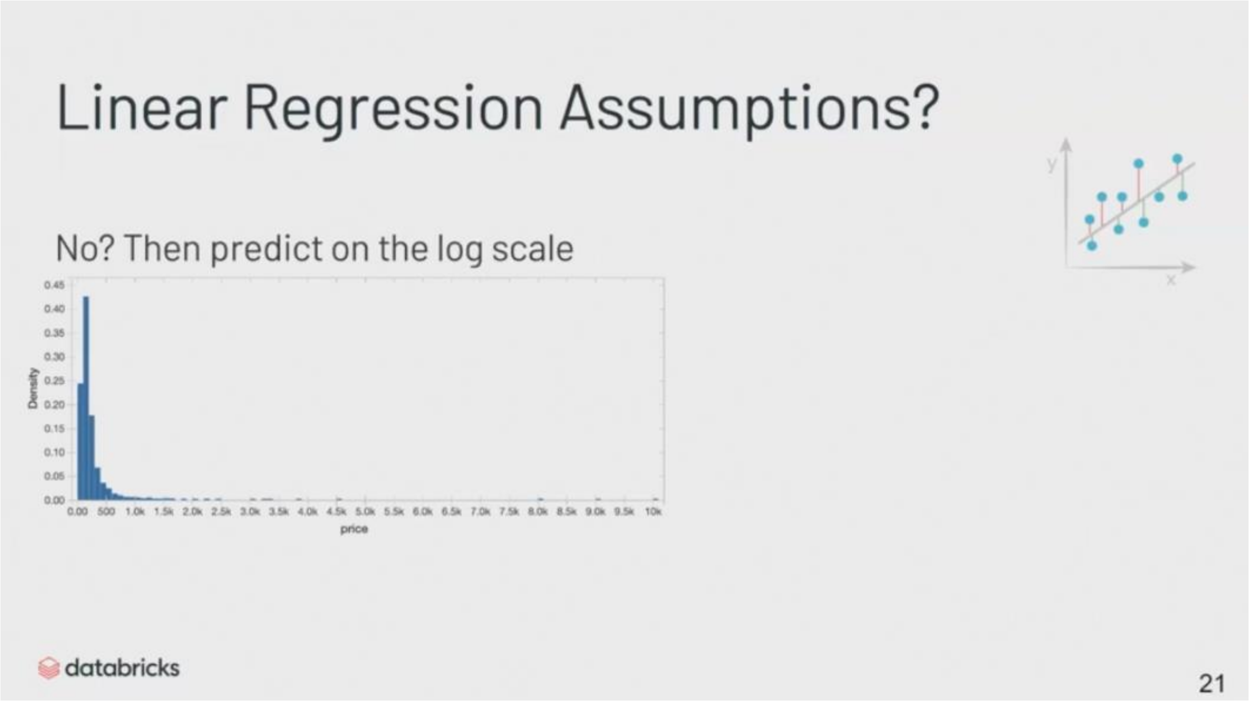
databricks

20



Linear Regression Assumptions?

databricks



Linear Regression Assumptions?

No? Then predict on the log scale

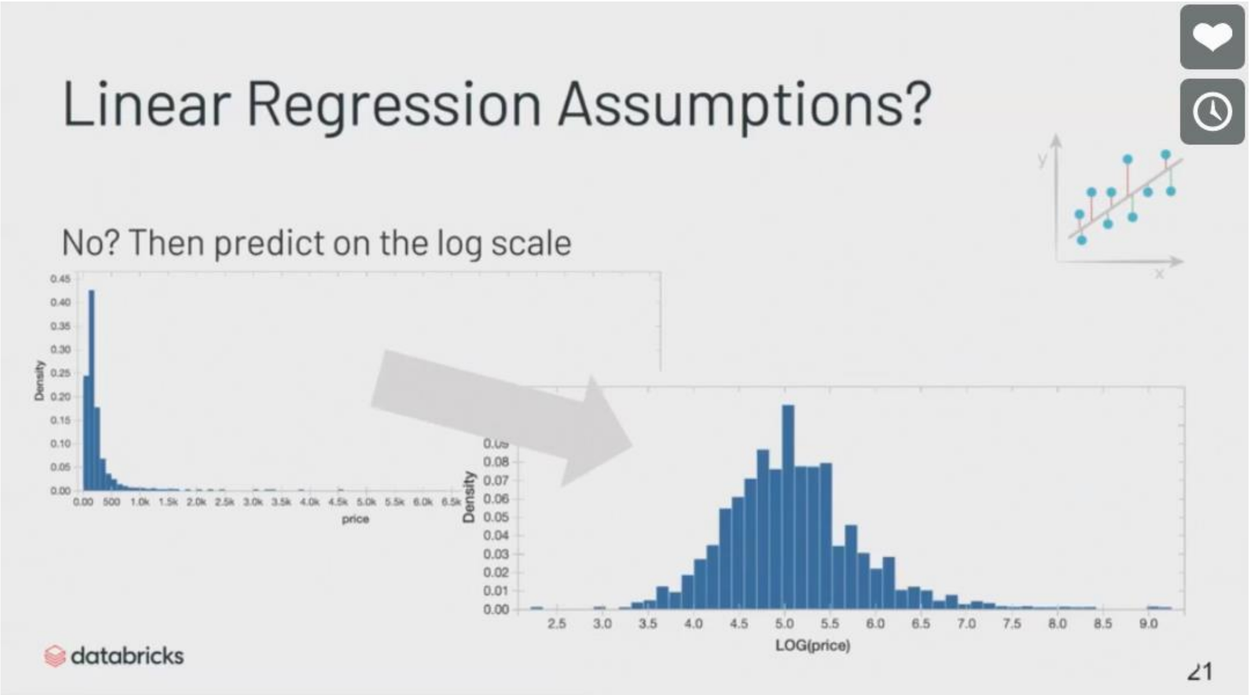
databricks

Linear regression assumptions:

Linear relationship between target and feature

Error is normally distributed

If data is skewed, apply log normal distribution

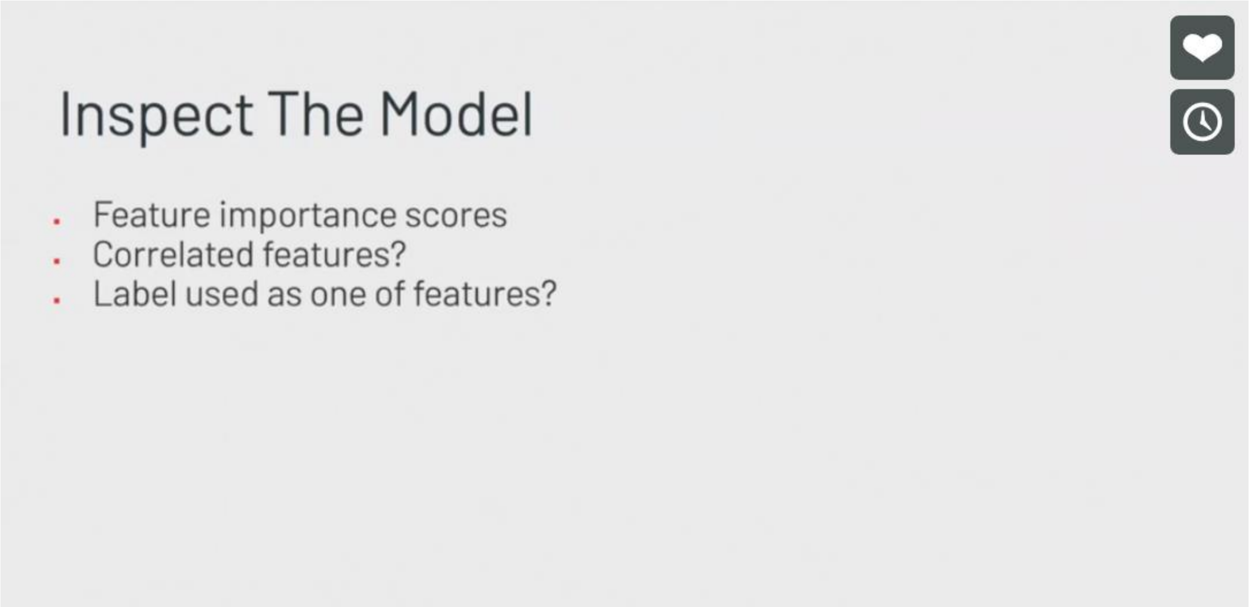


Linear Regression Assumptions?

No? Then predict on the log scale

databricks

21

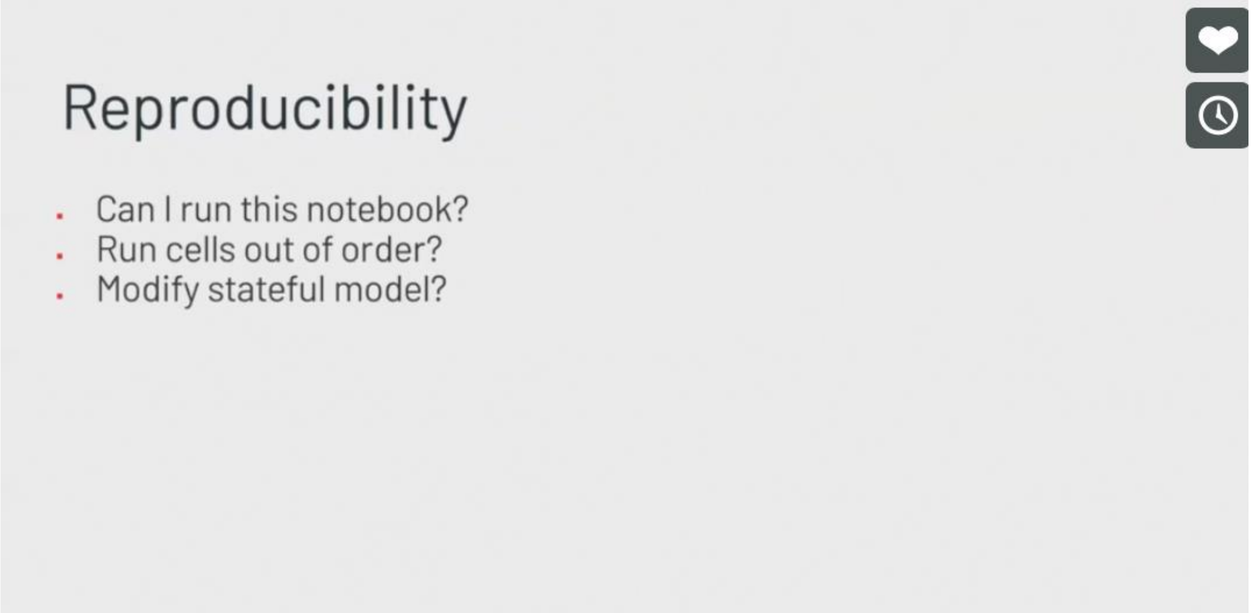


Inspect The Model

- Feature importance scores
- Correlated features?
- Label used as one of features?

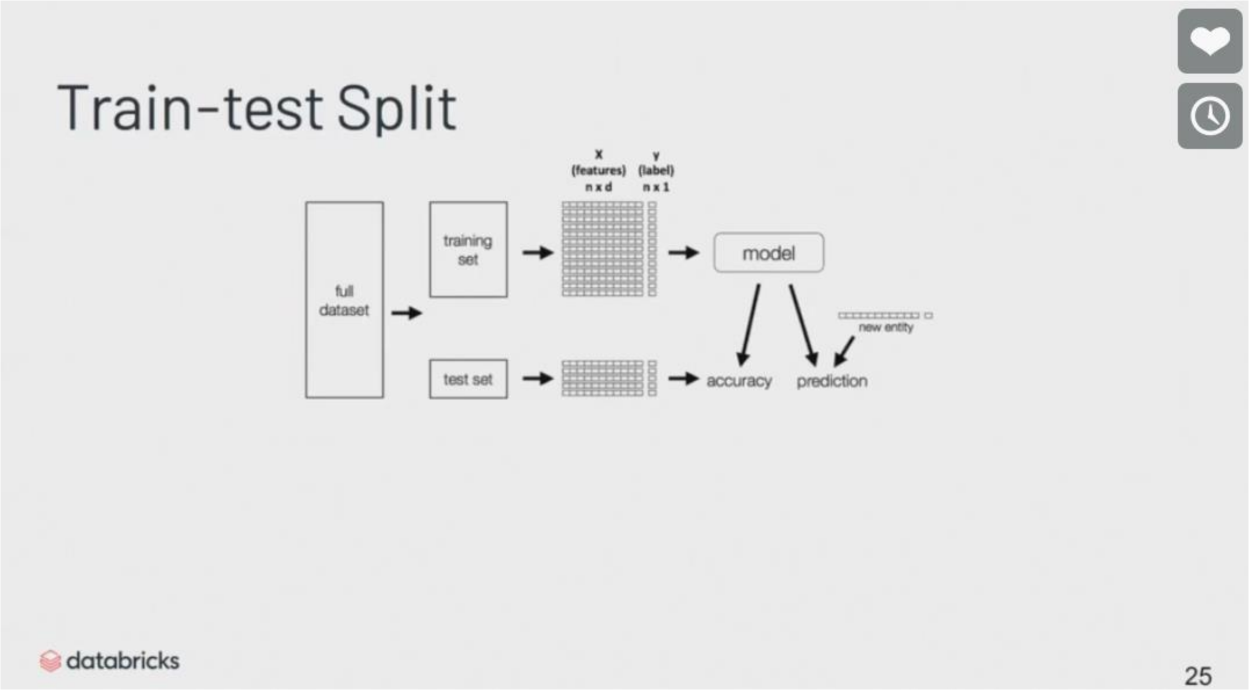
Model integrity + Performance

databricks



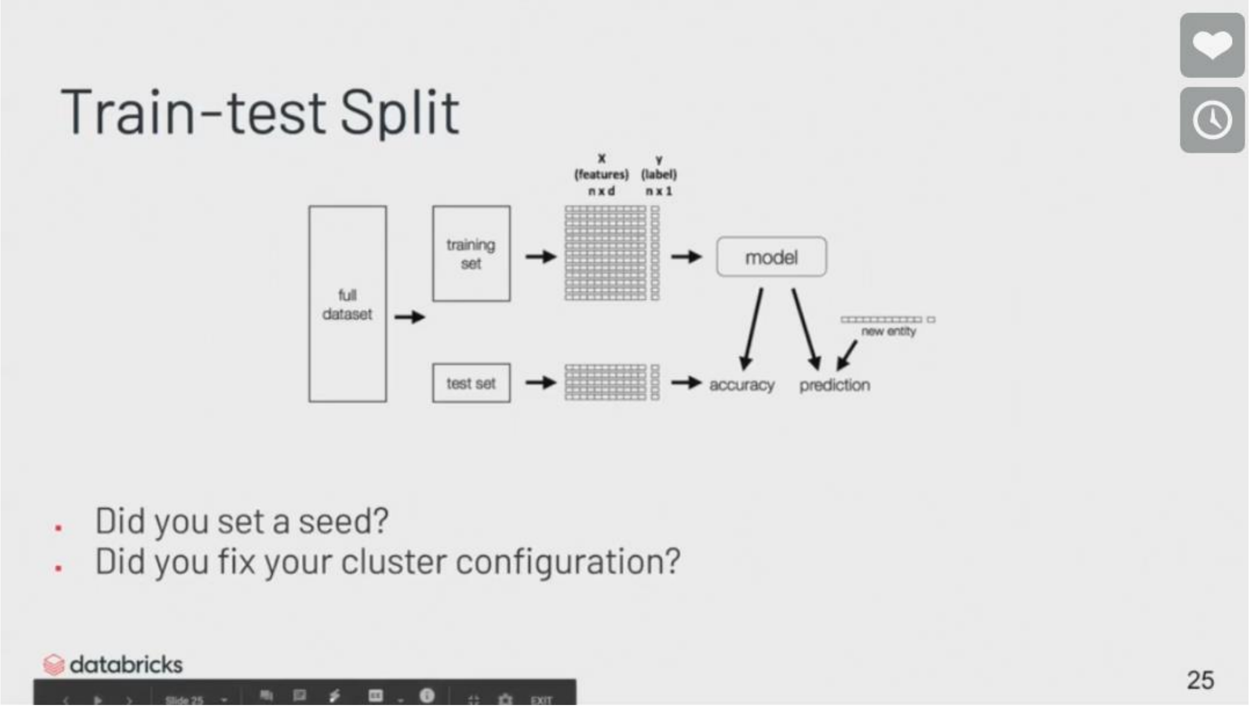
Reproducibility

- Can I run this notebook?
- Run cells out of order?
- Modify stateful model?



Train-test Split

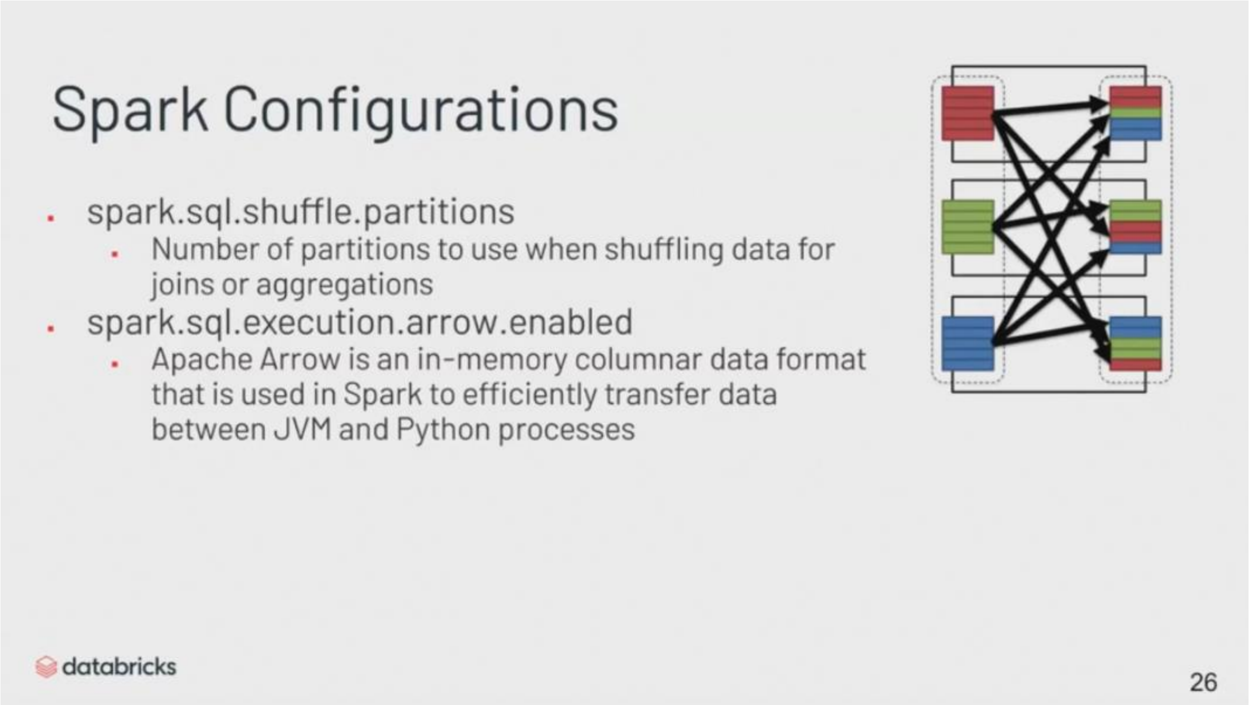
databricks



Train-test Split

- Did you set a seed?
- Did you fix your cluster configuration?

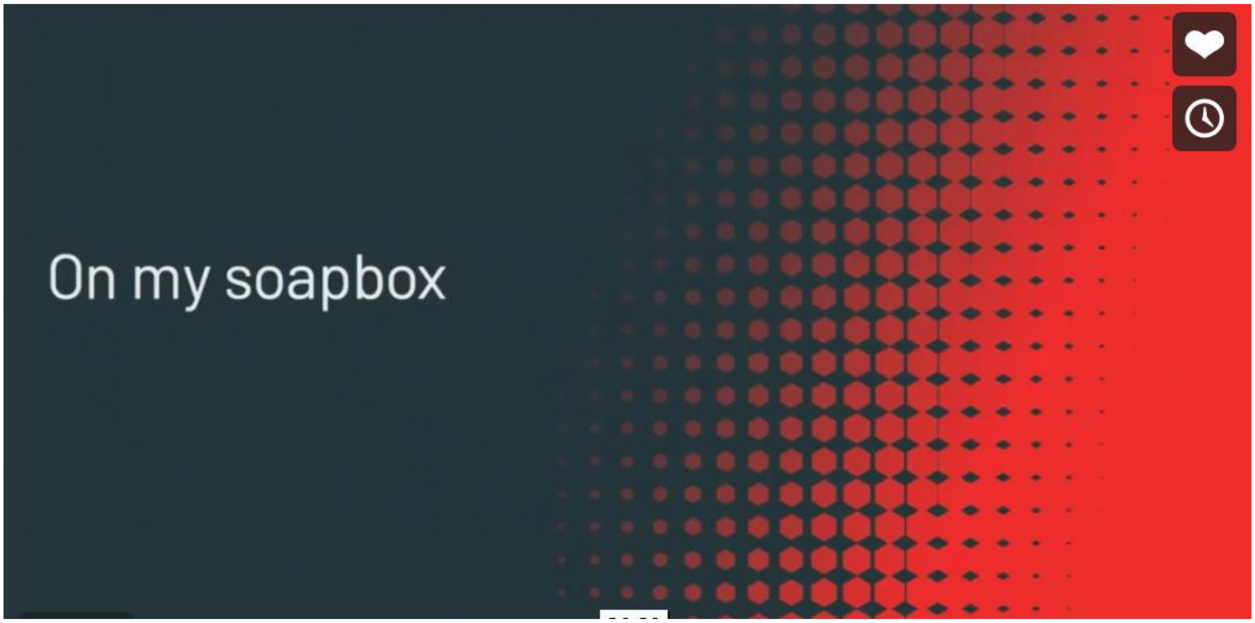
databricks



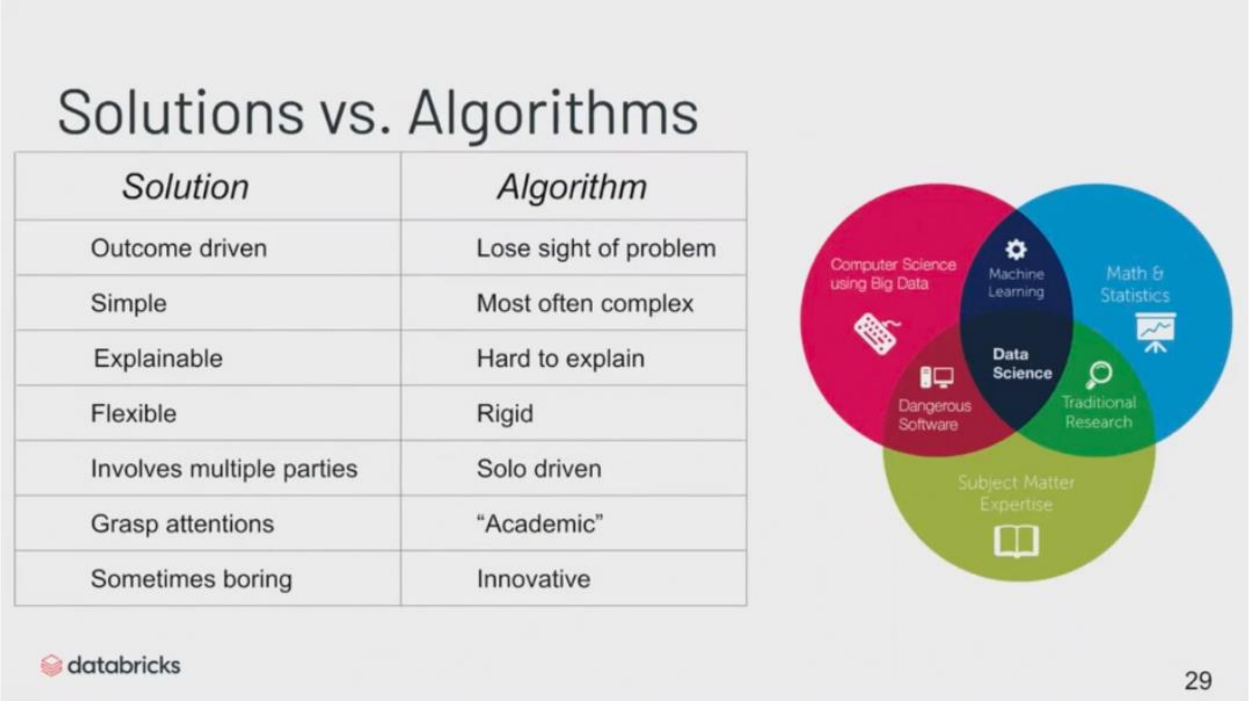
Spark Configurations

- spark.sql.shuffle.partitions
  - Number of partitions to use when shuffling data for joins or aggregations
- spark.sql.execution.arrow.enabled
  - Apache Arrow is an in-memory columnar data format that is used in Spark to efficiently transfer data between JVM and Python processes

Optimizing SparkML Pipeline Performance Demo

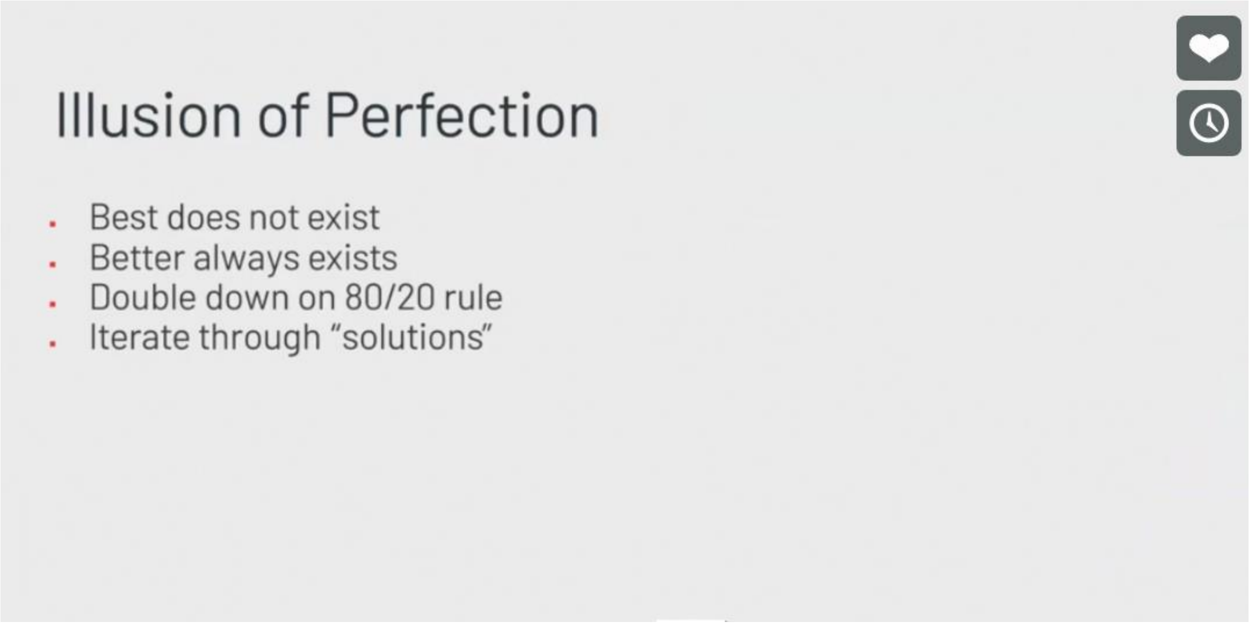


On my soapbox



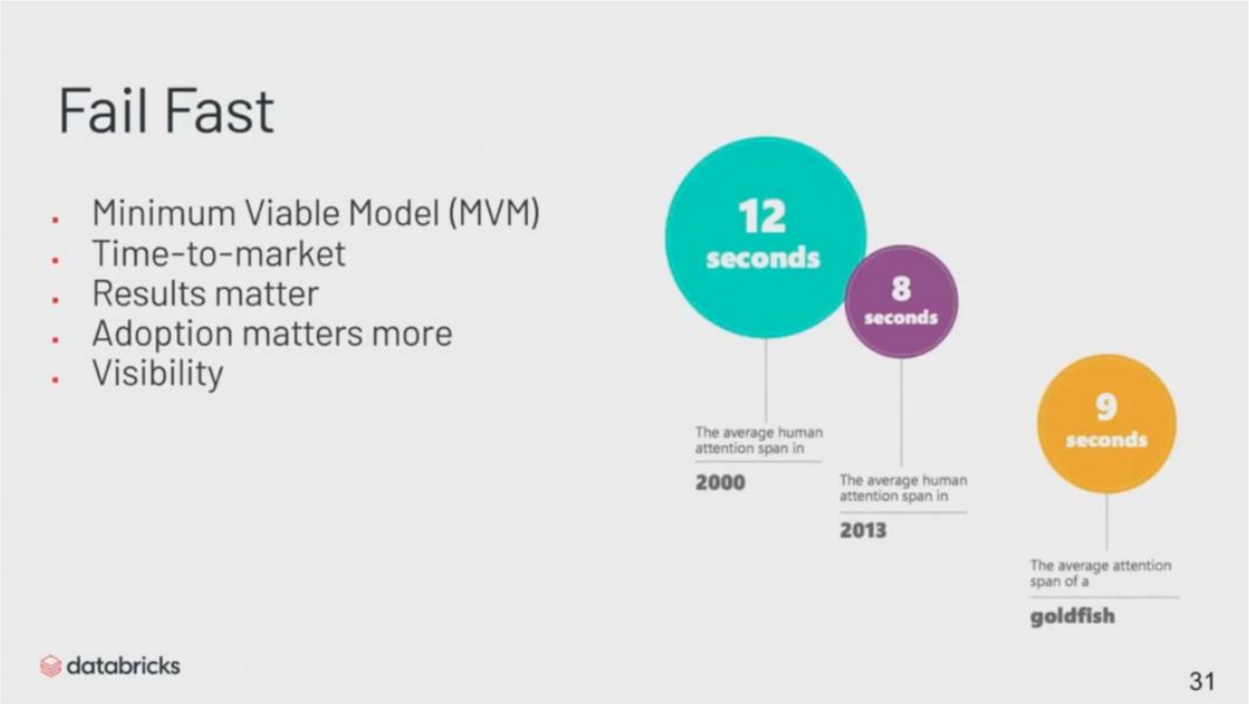
Solutions vs. Algorithms

Solution	*Algorithm*
Outcome driven	Lose sight of problem
Simple	Most often complex
Explainable	Hard to explain
Flexible	Rigid
Involves multiple parties	Solo driven
Grasp attentions	"Academic"
Sometimes boring	Innovative



Illusion of Perfection

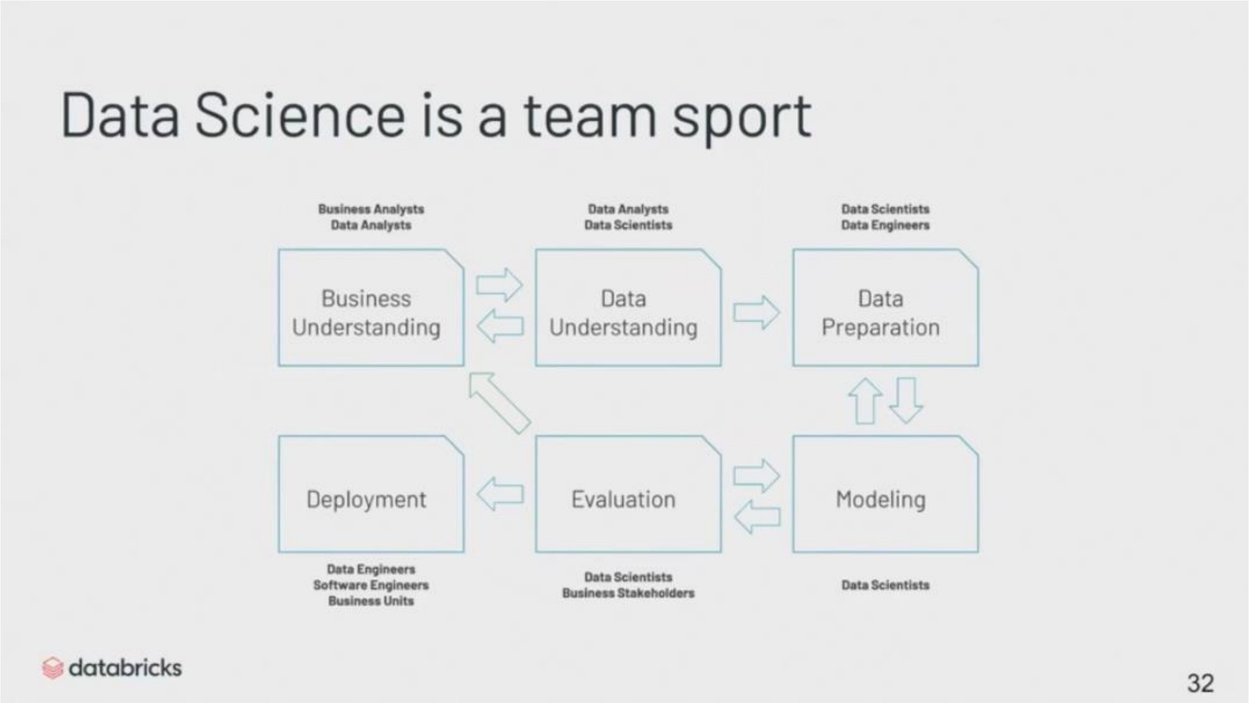
- Best does not exist
- Better always exists
- Double down on 80/20 rule
- Iterate through “solutions”



Fail Fast

- Minimum Viable Model (MVM)
- Time-to-market
- Results matter
- Adoption matters more
- Visibility

12 seconds
The average human attention span in 2000
8 seconds
The average human attention span in 2013
9 seconds
The average attention span of a goldfish



Data Science is a team sport

Business Analysts
Data Analysts

Business Understanding

Data Analysts
Data Scientists

Data Understanding

Data Scientists
Data Engineers

Data Preparation

Deployment

Data Engineers
Software Engineers
Business Units

Evaluation

Data Scientists
Business Stakeholders

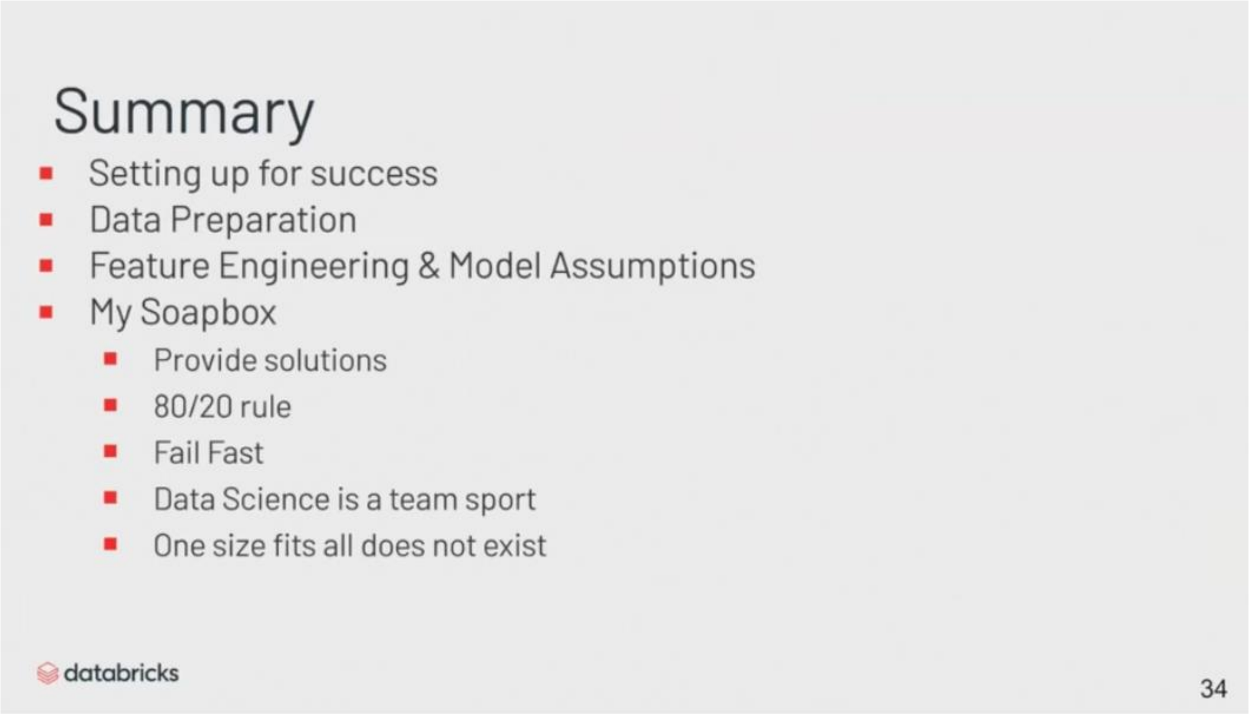
Modeling

Data Scientists

Theory of Everything

- Organizations are unique
- Problems are unique
- Datasets are unique
- DS life cycles are unique
- One-size-fits all solution does not exist (yet)!

I wonder if the doctor can give me something to stop me getting anxious about going to see the counsellor about my anxiety?
SURGERY



Summary

- Setting up for success
- Data Preparation
- Feature Engineering & Model Assumptions
- My Soapbox
  - Provide solutions
  - 80/20 rule
  - Fail Fast
  - Data Science is a team sport
  - One size fits all does not exist