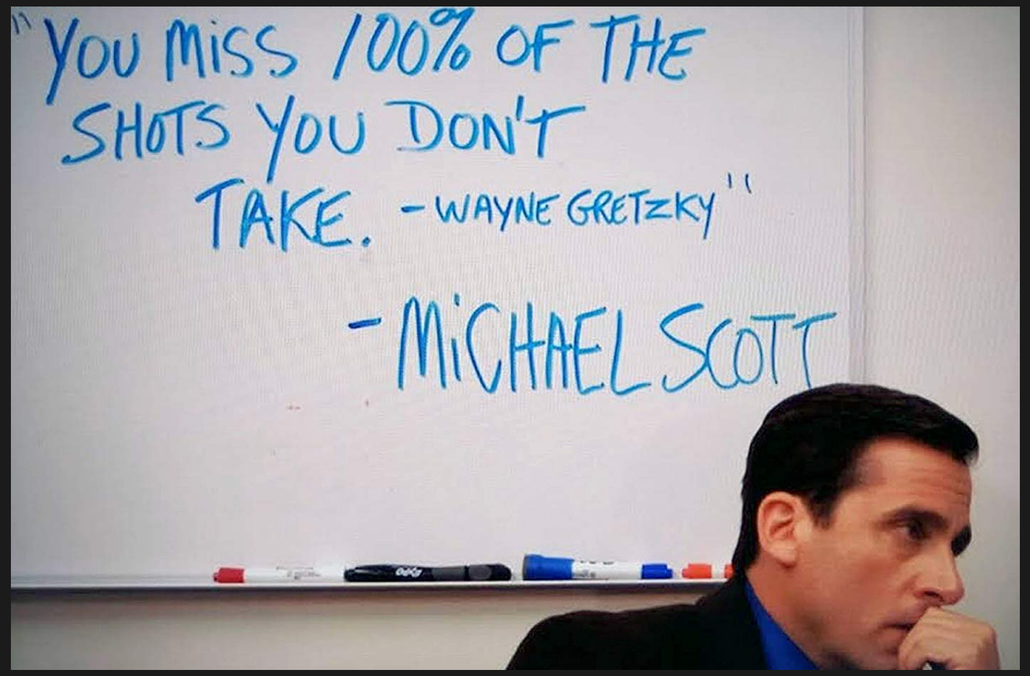
EFFICIENT DATA-DRIVEN VISUALIZATION RECOMMENDATIONS TO SUPPORT VISUAL ANALYTICS

SeeDB

- Manasi Vartak, Sajjadur Rahman, Samuel Madden, Aditya Parameswaran, Neoklis Polyzotis

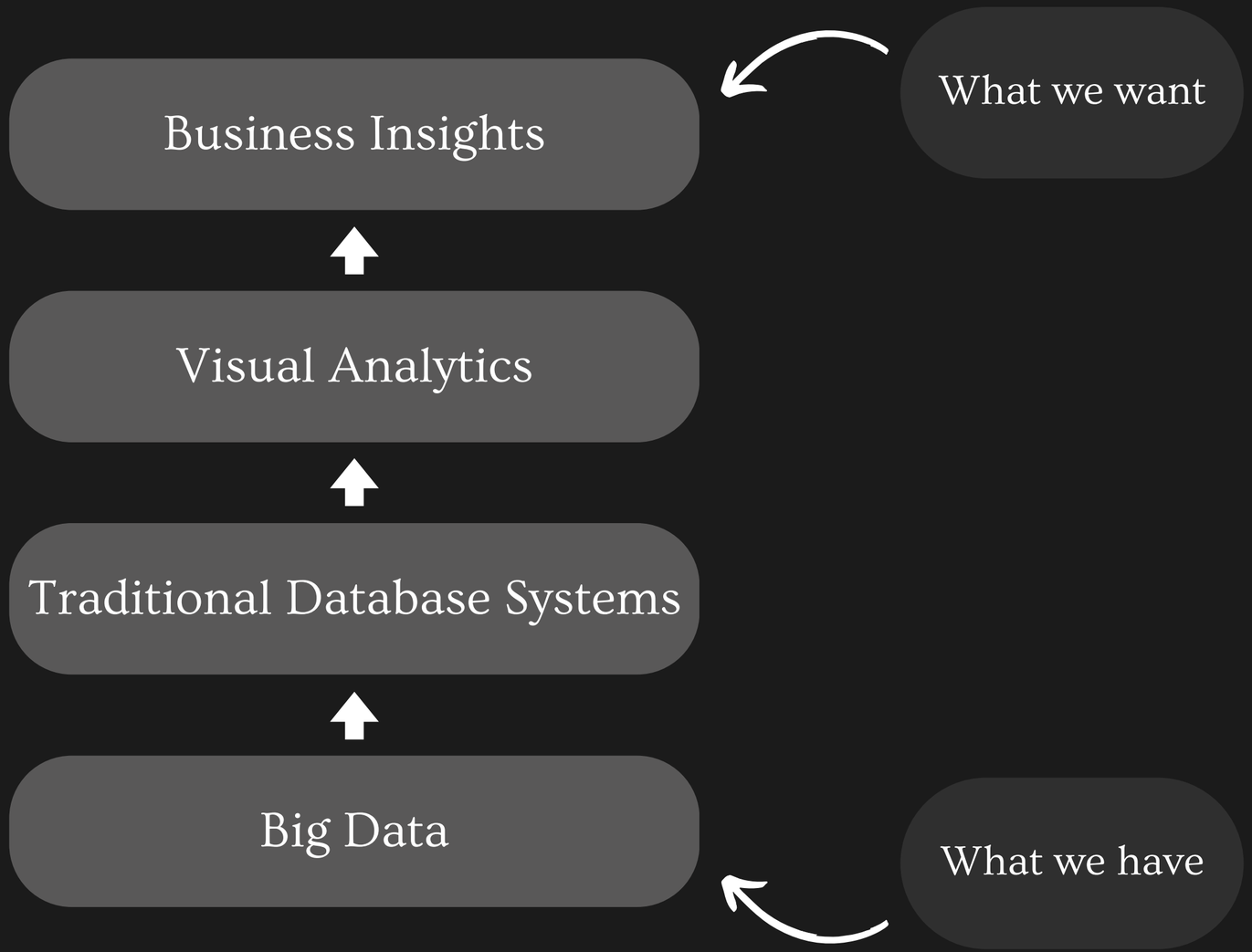
- Ankita Shanbhag



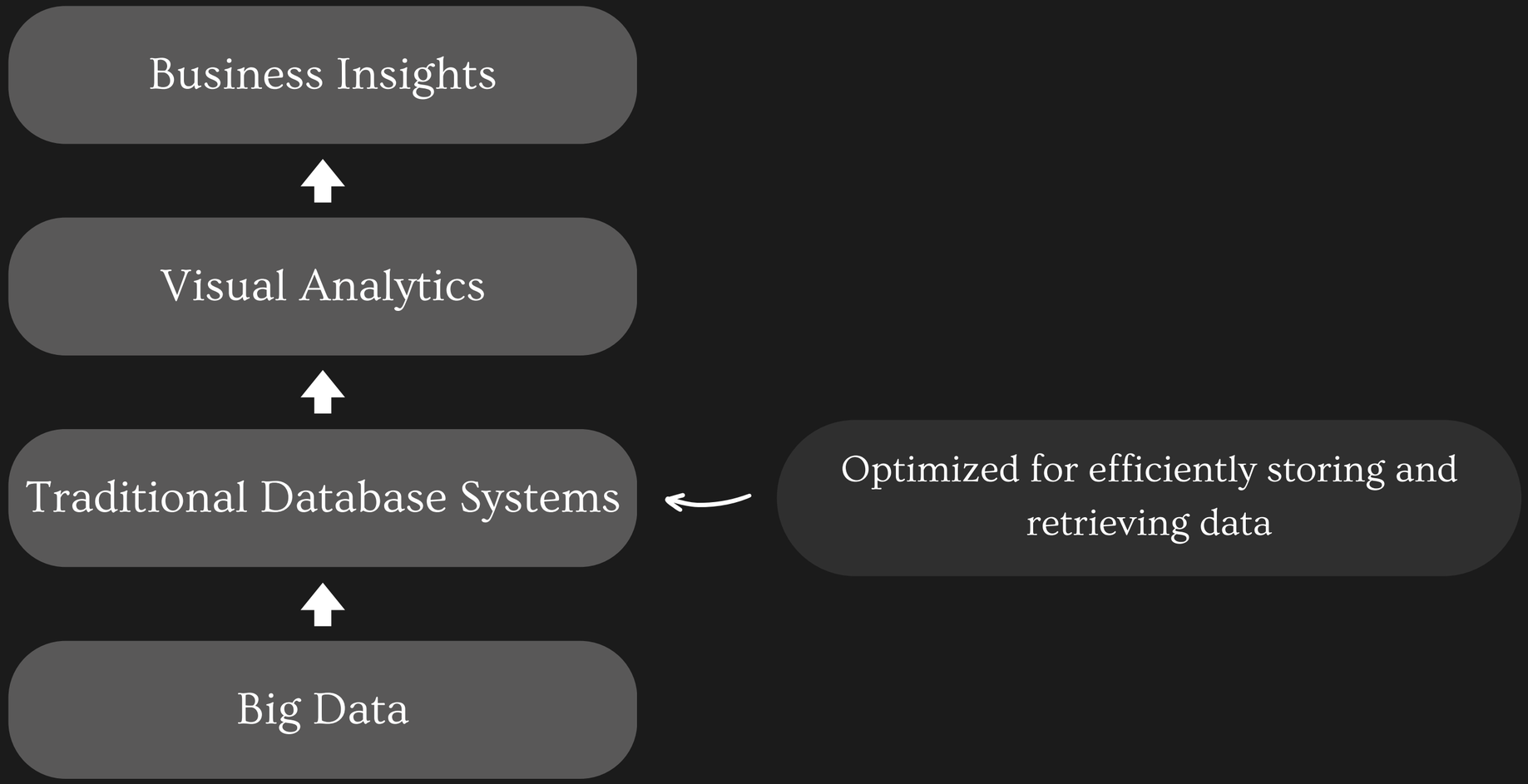
Outline

- The Problem
- SeeDB - the Solution
- SeeDB Architecture
- Utility
- Optimization through sharing
- Optimization through pruning
- Evaluation
- User Study
- Next Steps

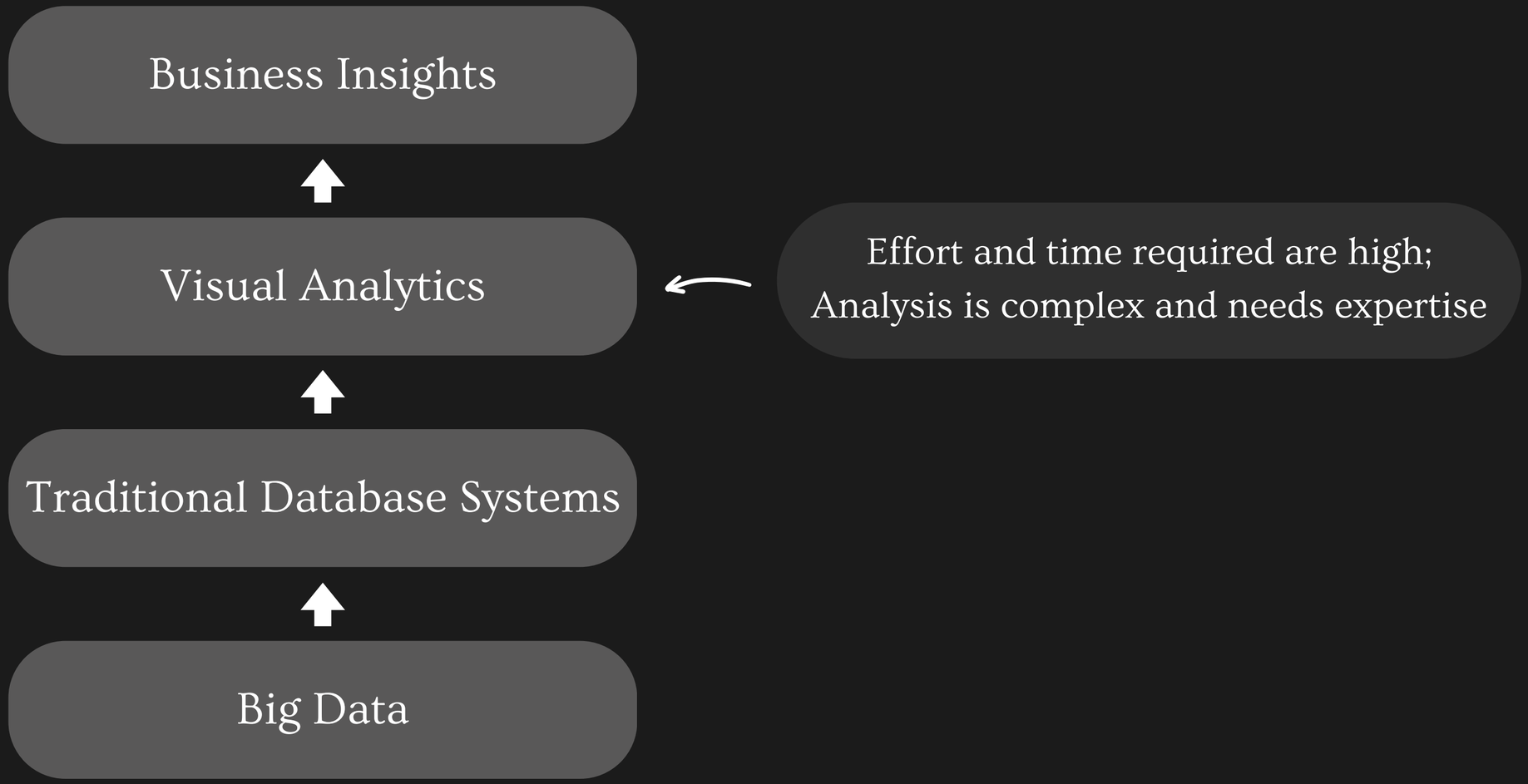
The Problems and Limitations of Traditional Database Systems



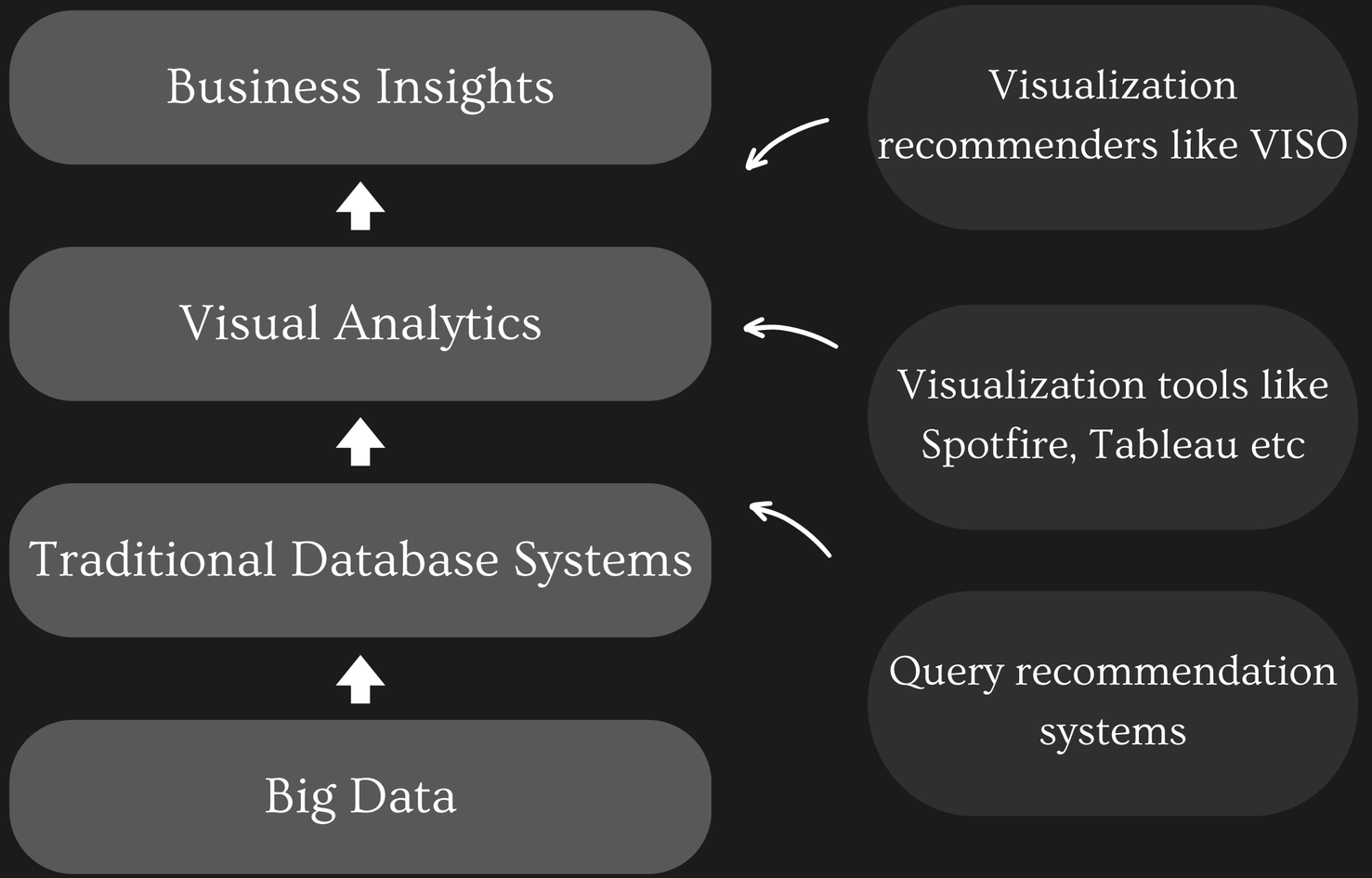
The Problems and Limitations of Traditional Database Systems



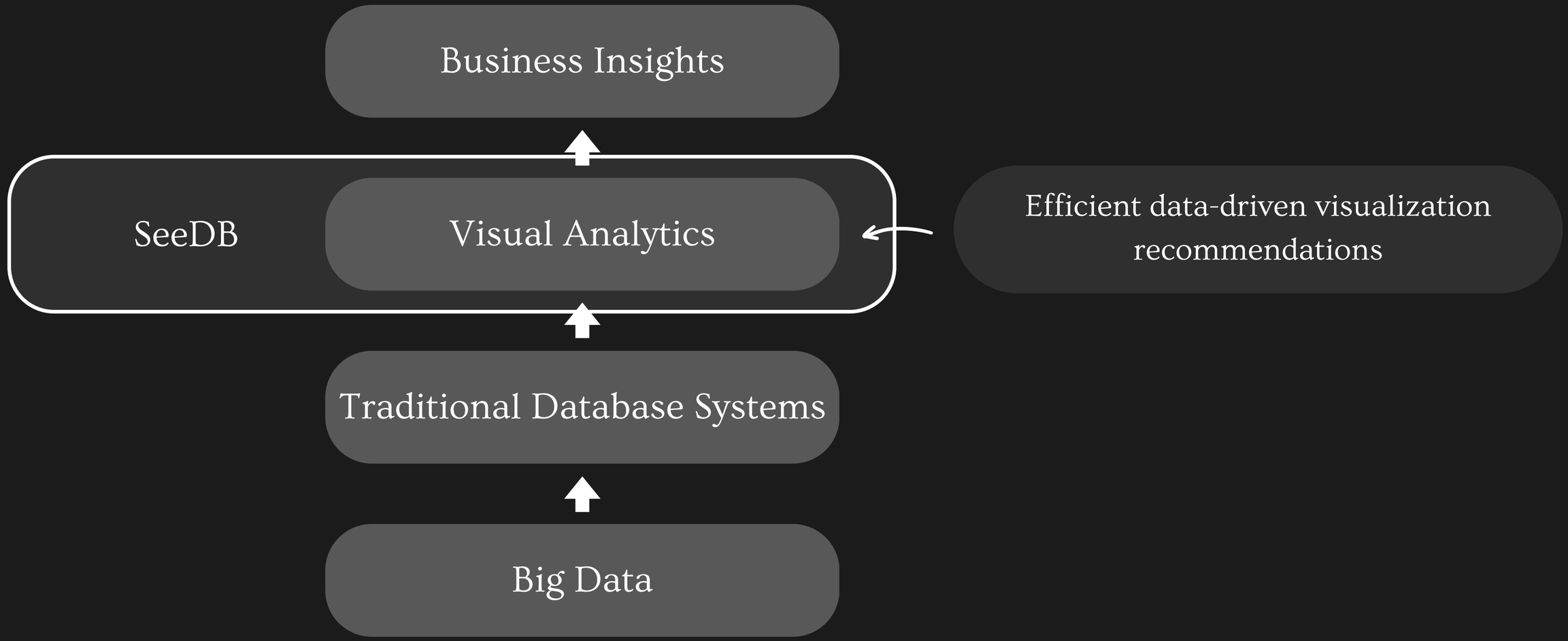
The Problems and Limitations of Traditional Database Systems



Currently available solutions (in 2015)



A Middleware Layer for Data-Driven Visualization Recommendations




Selection Criteria
SeeDB Client
View and interact with visualizations
Query
SeeDB Server
Most interesting views
Database Management System

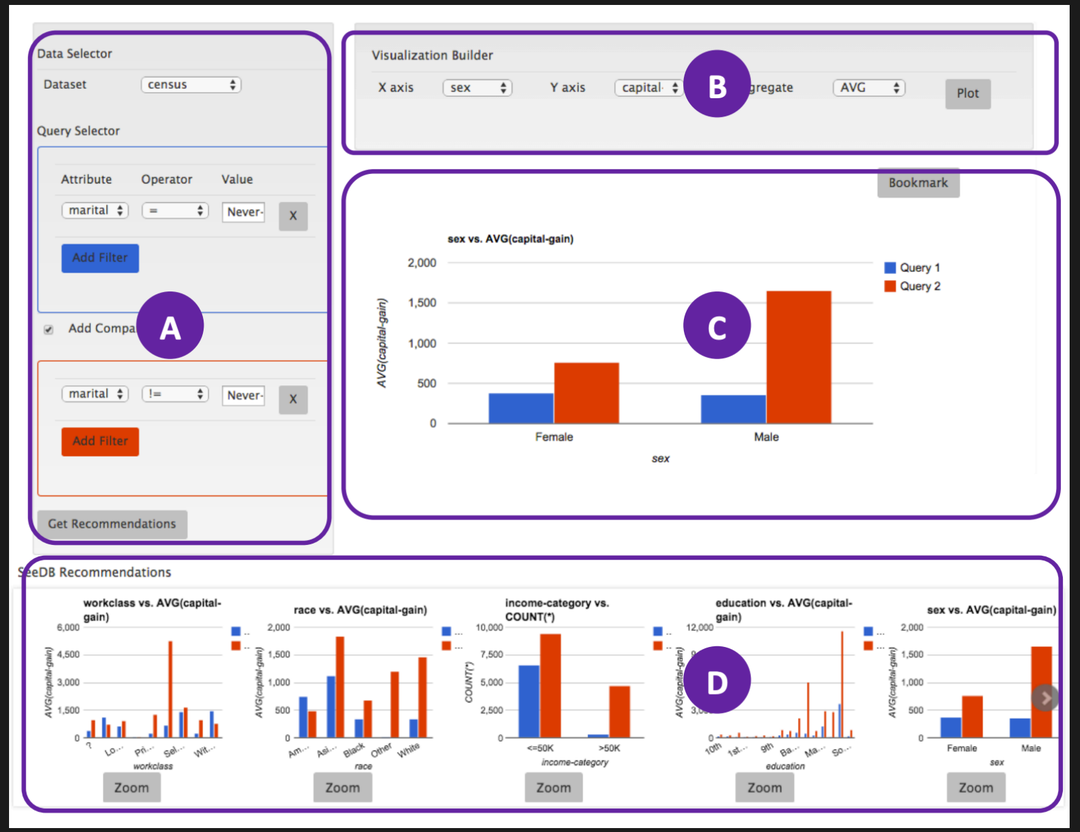
Selection Criteria

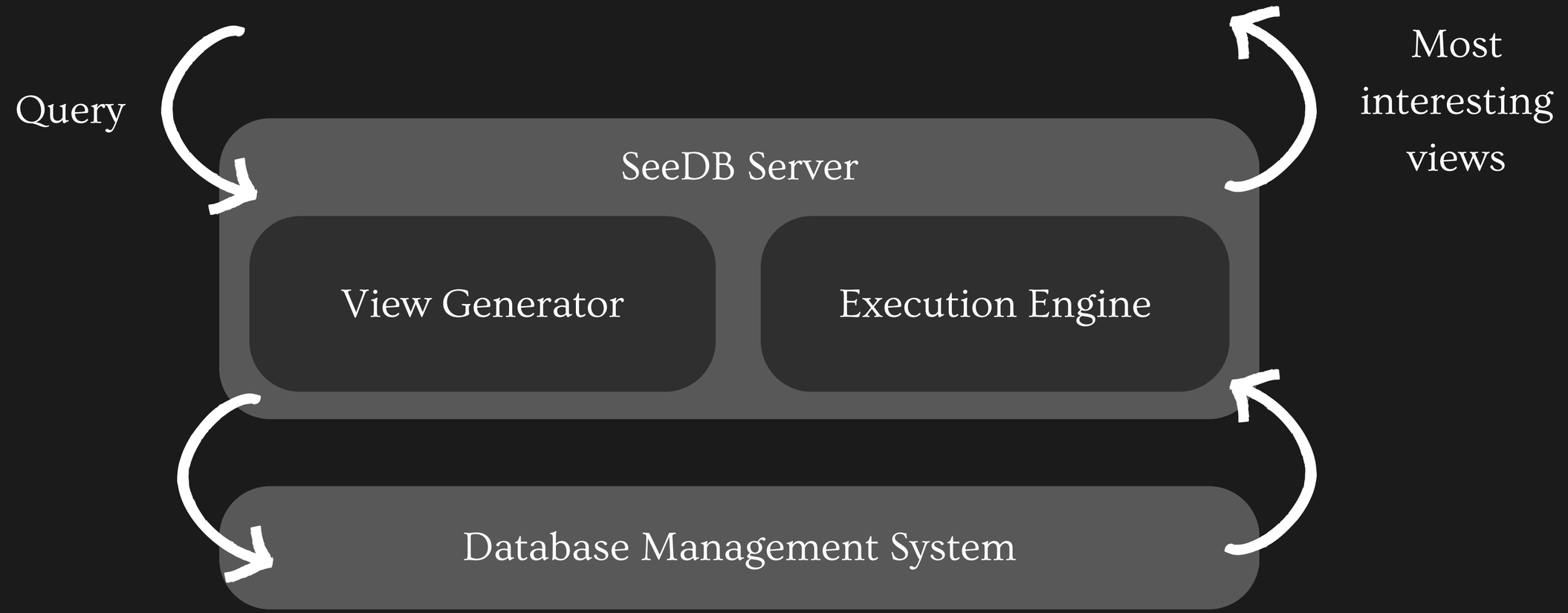
SeeDB Client

Query Builder

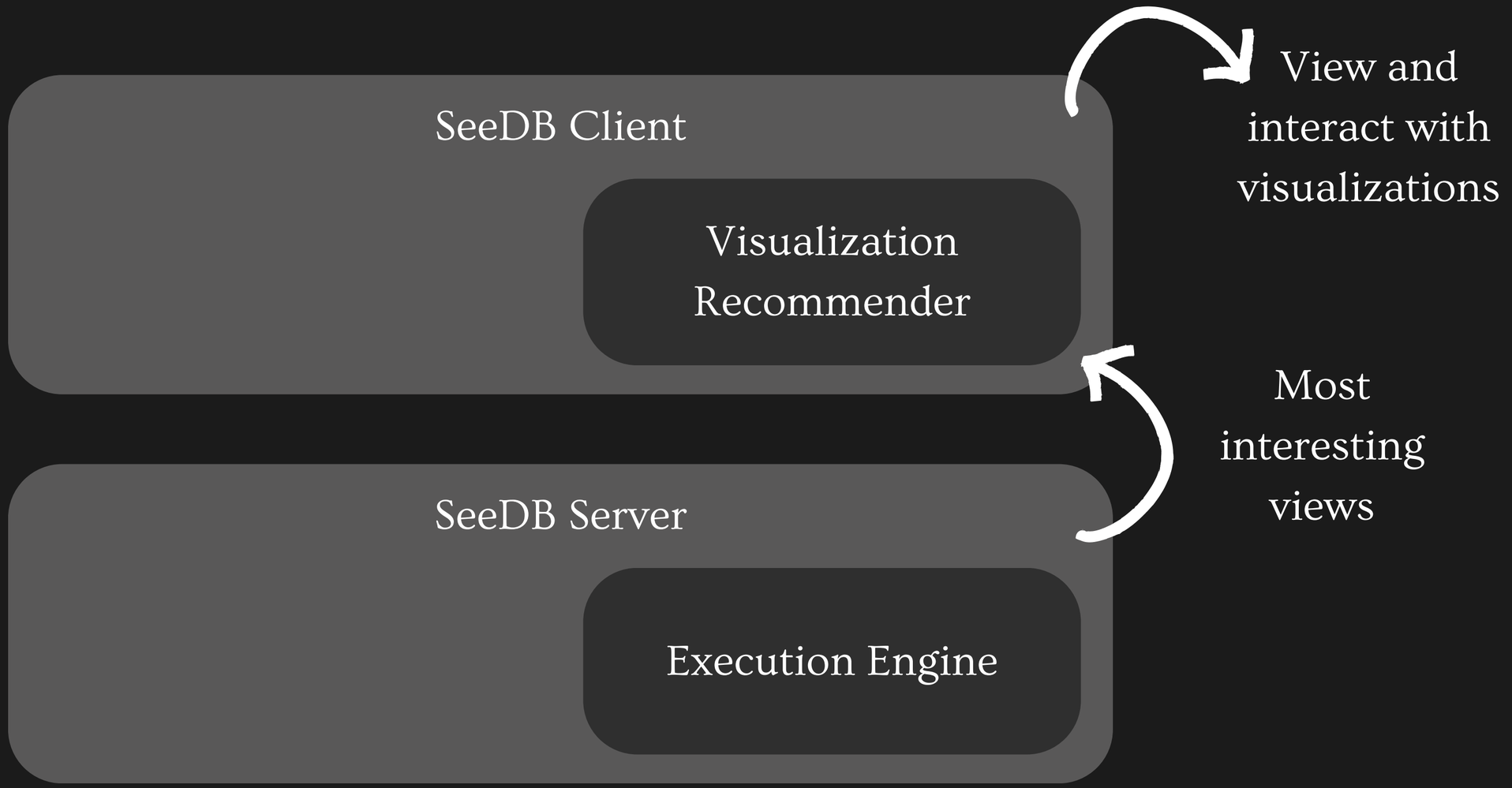
Visualization Recommender

View and interact with visualizations



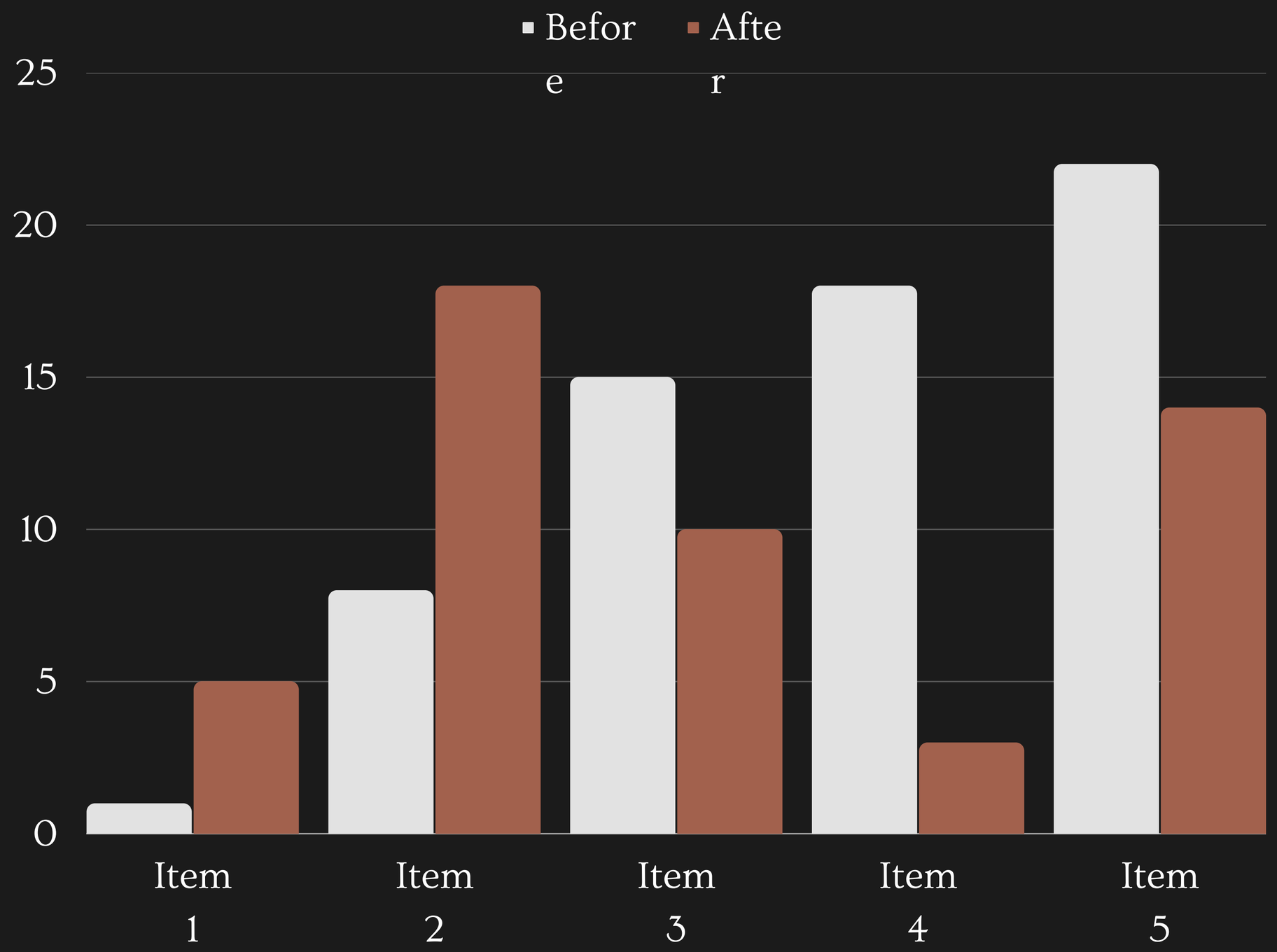

Query
SeeDB Server
View Generator
Execution Engine
Database Management System
Most interesting views

How do you define ‘interestingness’ or utility?



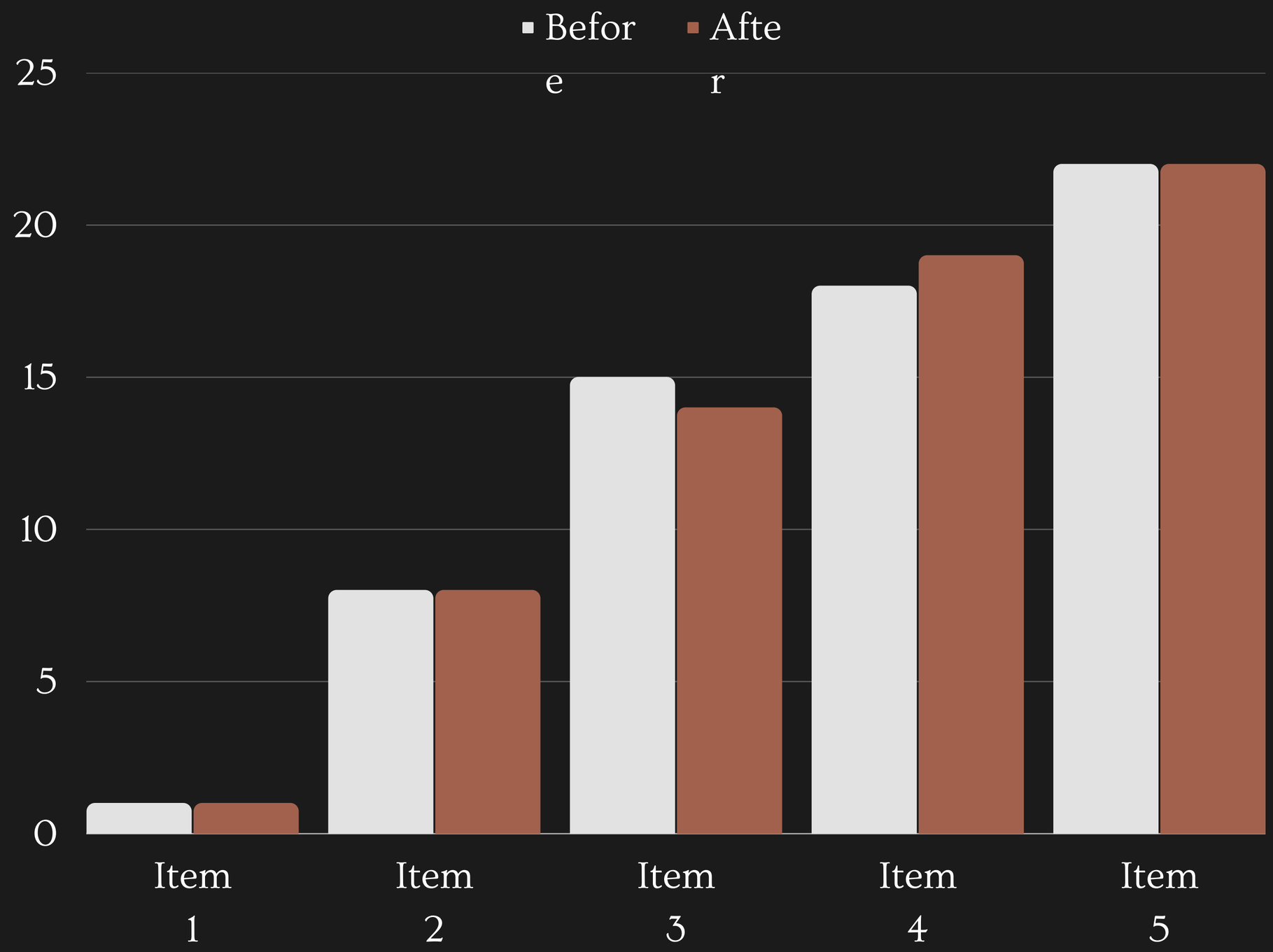
Option 1

Evaluating before vs after



Option 2

Evaluating before vs after



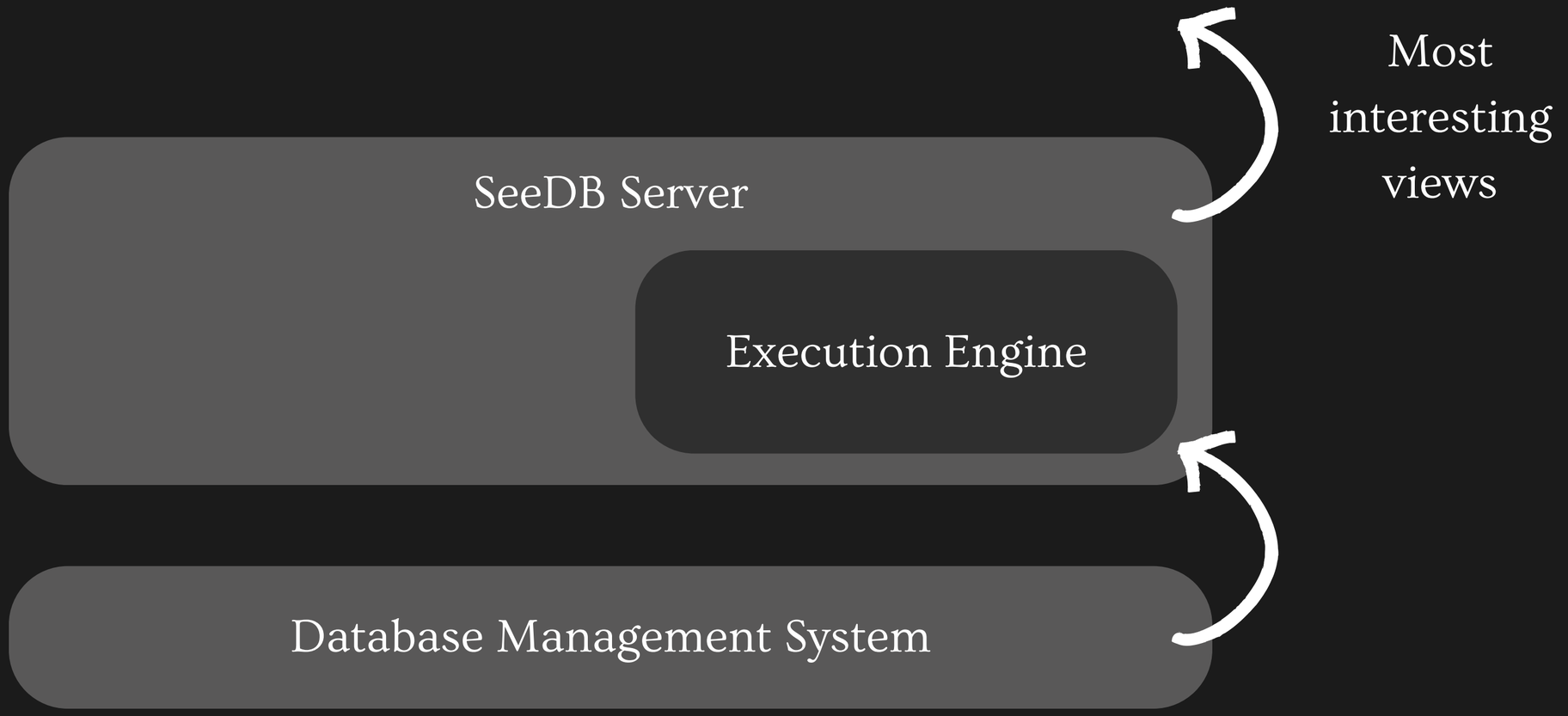
How do you define ‘interestingness’ or utility?

We define a proxy metric based on ‘deviation’ from the reference value

Other alternatives for a utility function -

- Outlier detection
- Correlation
- Similarity
- Presence of clusters
- Presence of patterns

How do you optimize executions?



How do you optimize executions?

Sharing

Sharing computational resources

Pruning

Pruning low-utility views

How do you optimize executions by sharing?

Combine multiple aggregate view queries

Combine multiple GROUP BYs

Combine target and reference queries

Parallelize query execution

How do you optimize executions by sharing?

Combine multiple aggregate view queries

```
SELECT a1, a2
FROM table
GROUP BY a1
```

```
SELECT a1, SUM(m1)
FROM table
GROUP BY a1
```

```
SELECT a1, AVG(m2)
FROM table
GROUP BY a1
```

```
SELECT a1, a2, SUM(m1), AVG(m2)
FROM table
GROUP BY a1
```

How do you optimize executions by sharing?

```
SELECT *
FROM table
GROUP BY a1
```

```
SELECT *
FROM table
GROUP BY a2
```

```
SELECT *
FROM table
GROUP BY a3
```

Combine multiple GROUP BYs

```
SELECT *
FROM table
GROUP BY a1, a2, a3
```

How do you optimize executions by sharing?

```
SELECT *
FROM table
WHERE option = ‘option1’
GROUP BY a1
```

```
SELECT *
FROM table
WHERE option <> ‘option1’
GROUP BY a1
```

Combine target and reference queries

```
SELECT *,
CASE IF option = ‘option1’ THEN 1 ELSE 0
END AS g1,
FROM table
GROUP BY a1, g1
```

How do you optimize executions by sharing?

Parallelize query execution

How do you optimize executions by pruning?

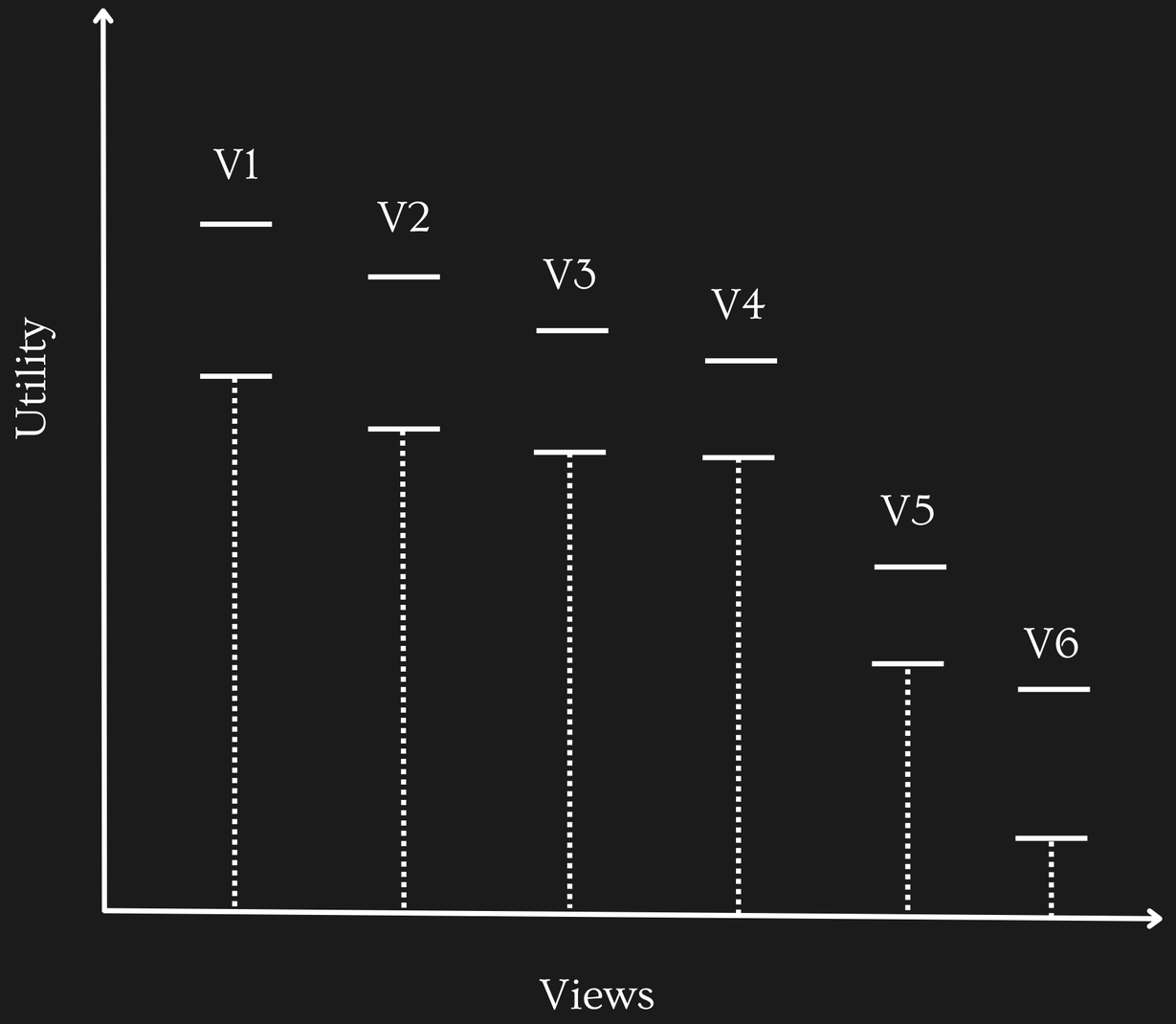
Selecting top-k views through pruning

Confidence interval-based pruning

Multi-armed bandit pruning

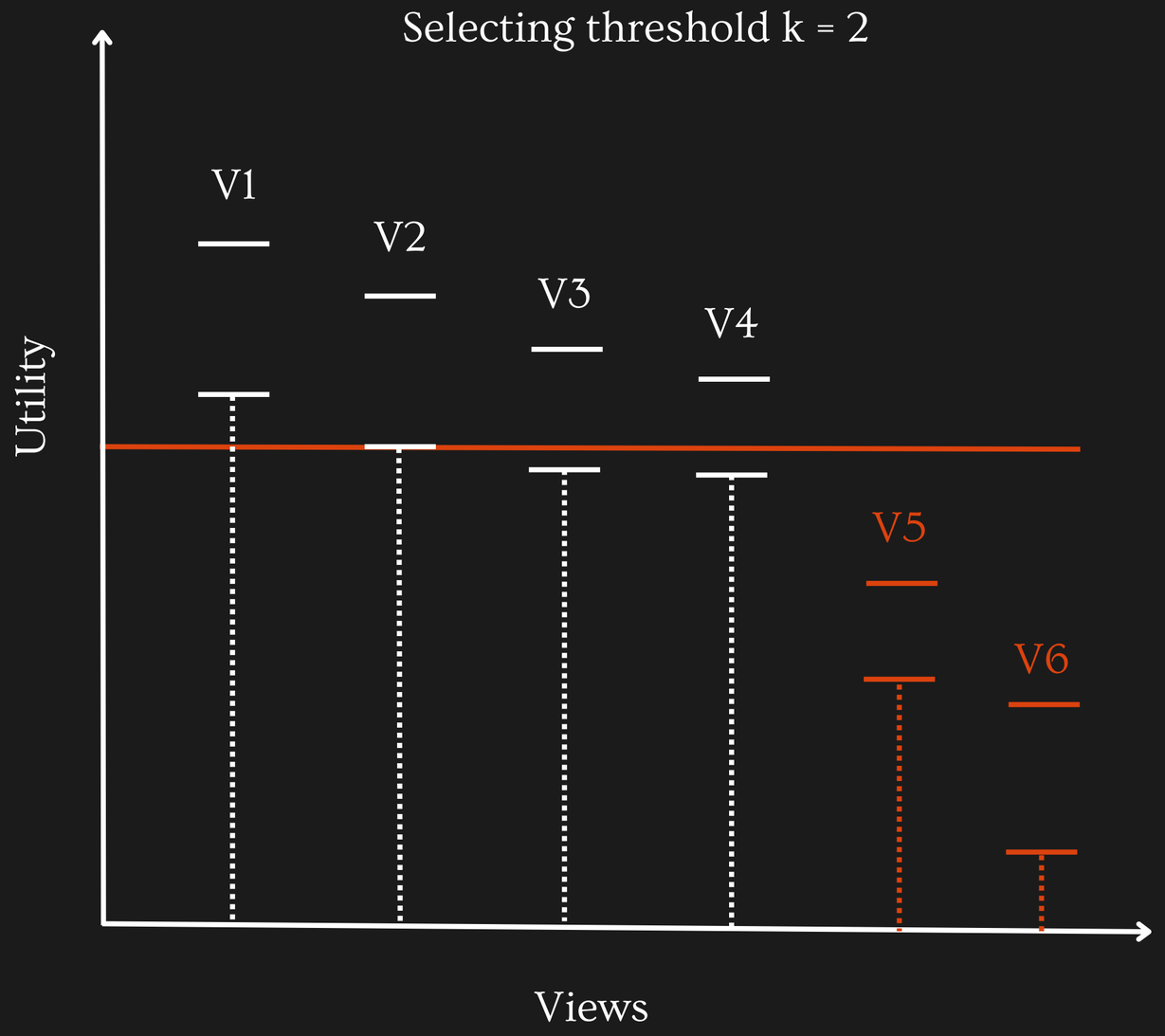
How do you optimize executions by pruning?

Confidence interval-based pruning



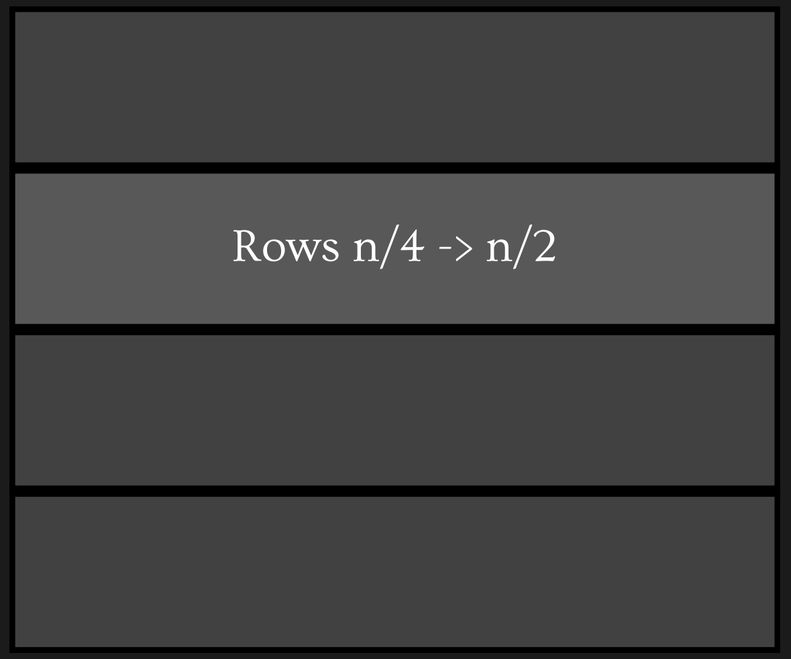
How do you optimize executions by pruning?

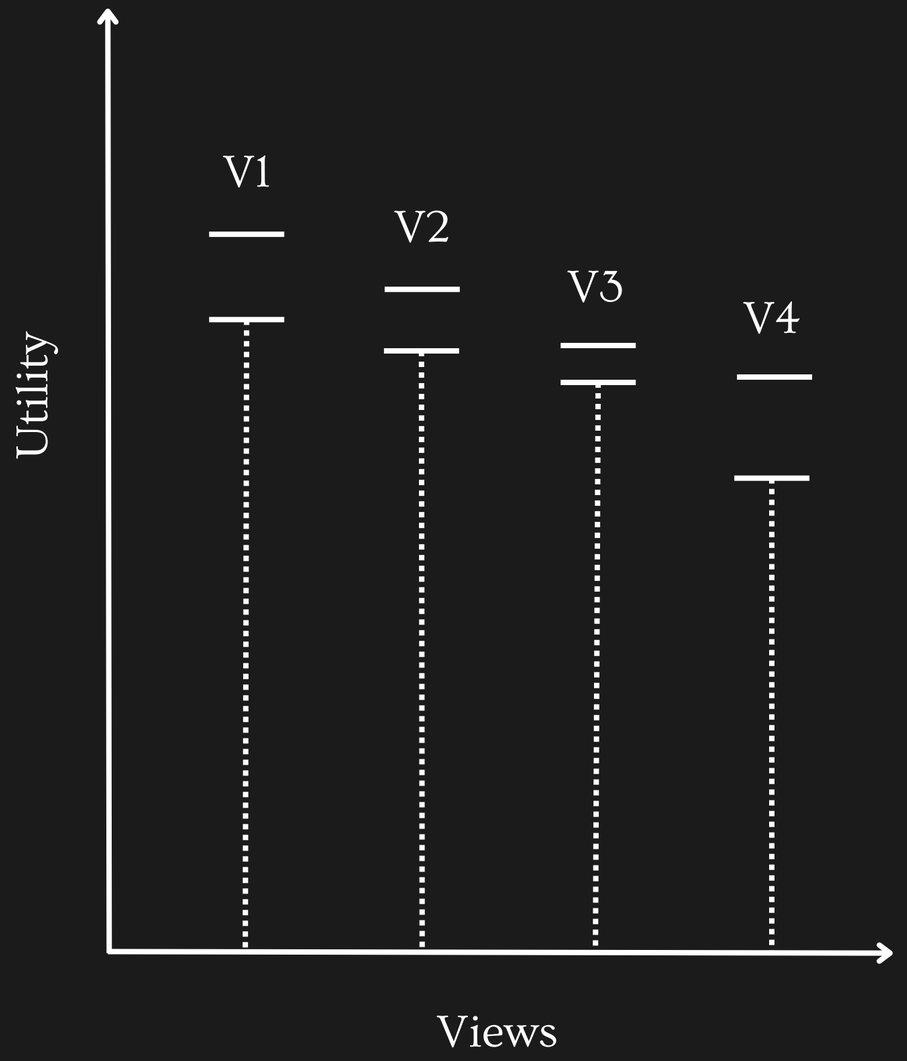
Confidence interval-based pruning



How do you optimize executions by pruning?

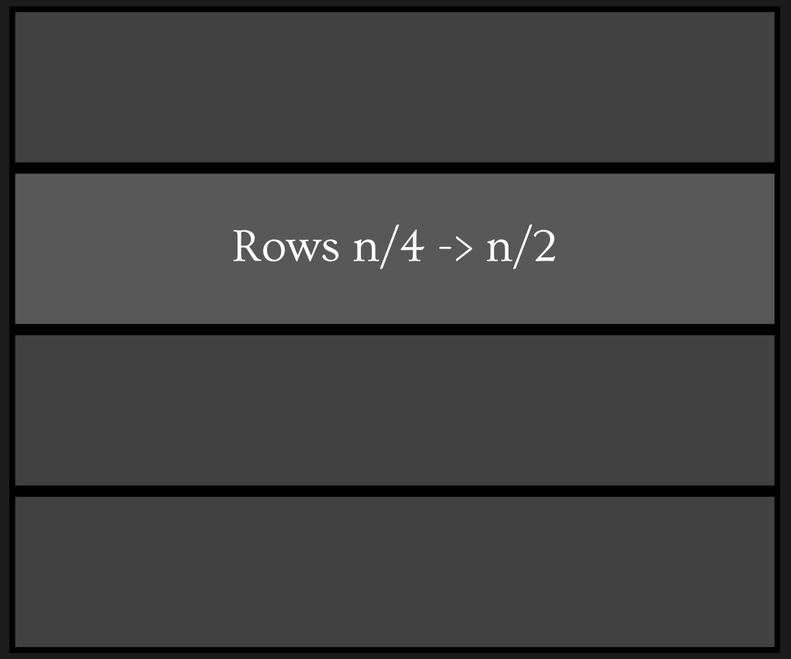
Confidence interval-based pruning

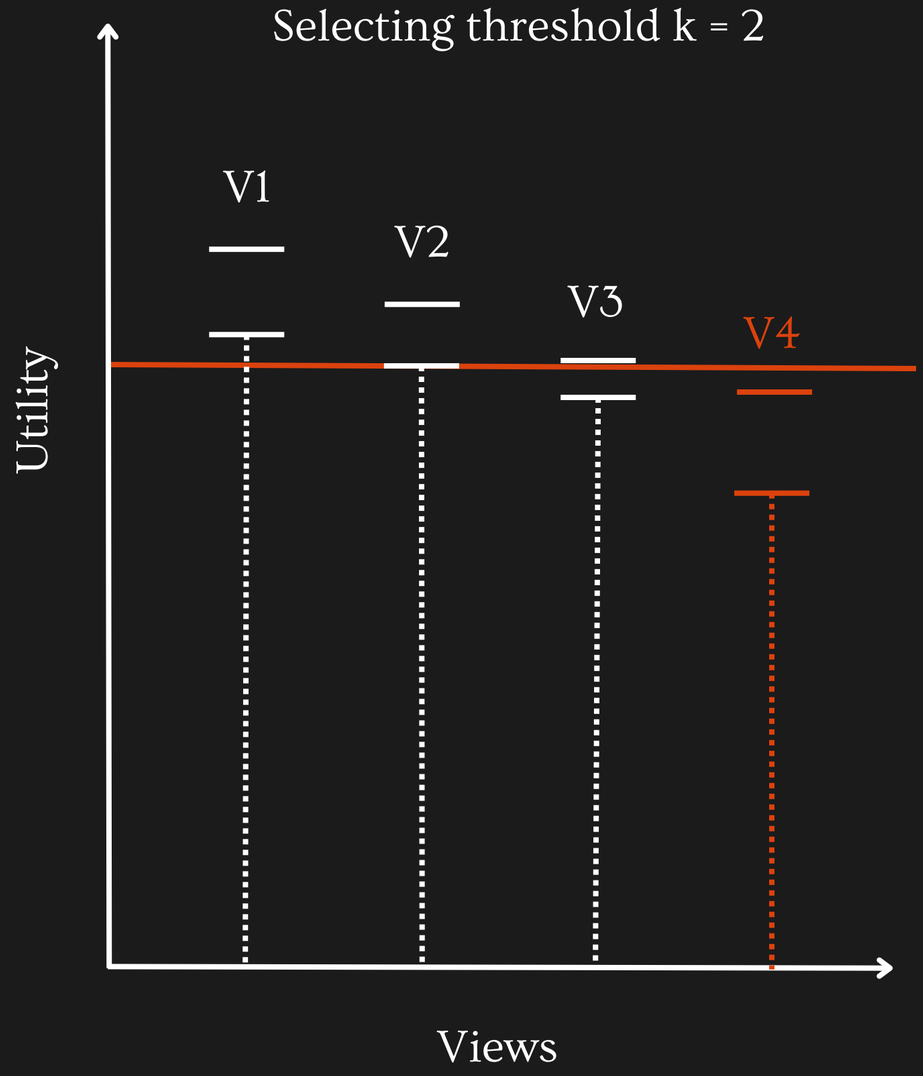




How do you optimize executions by pruning?

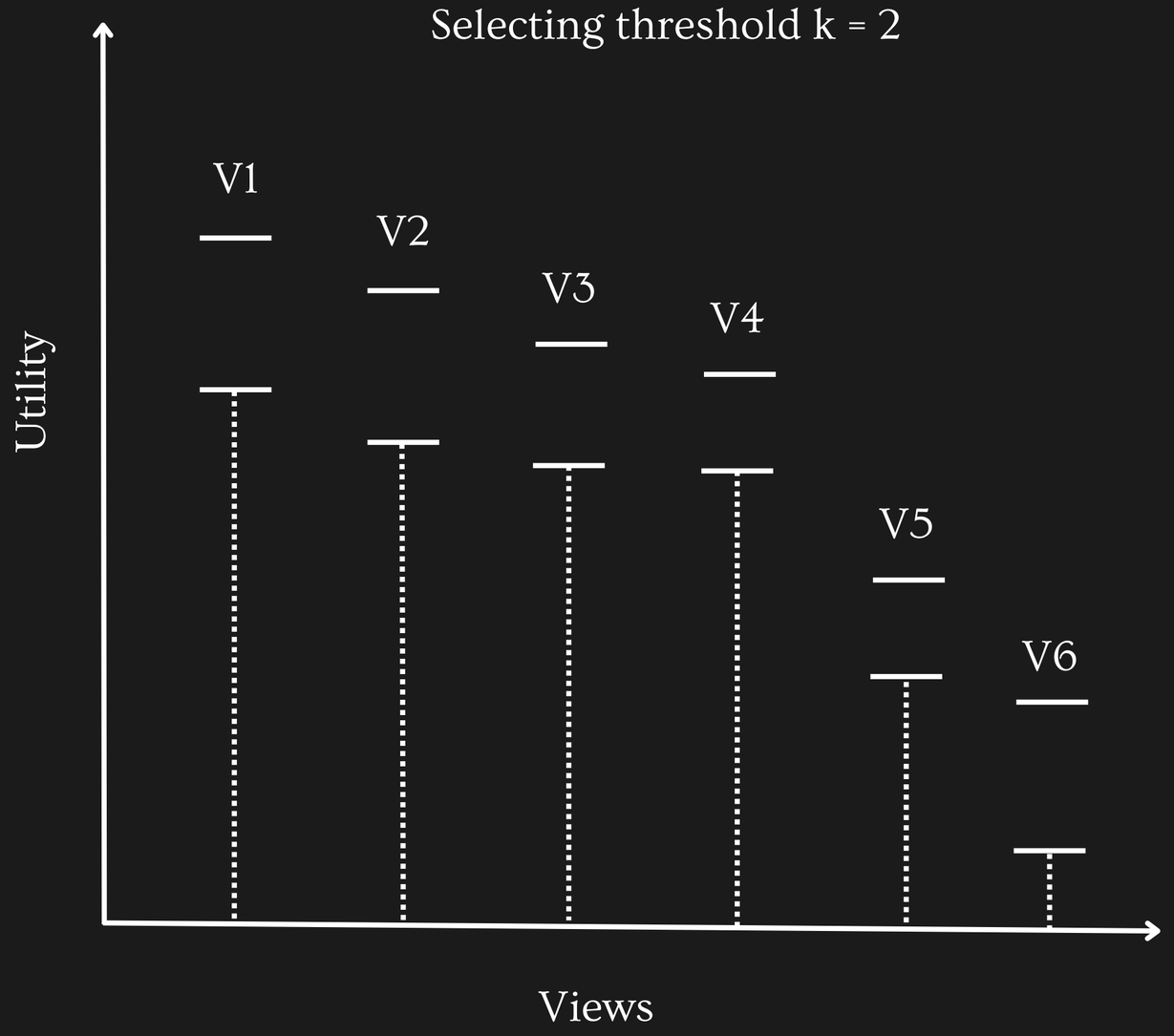
Confidence interval-based pruning





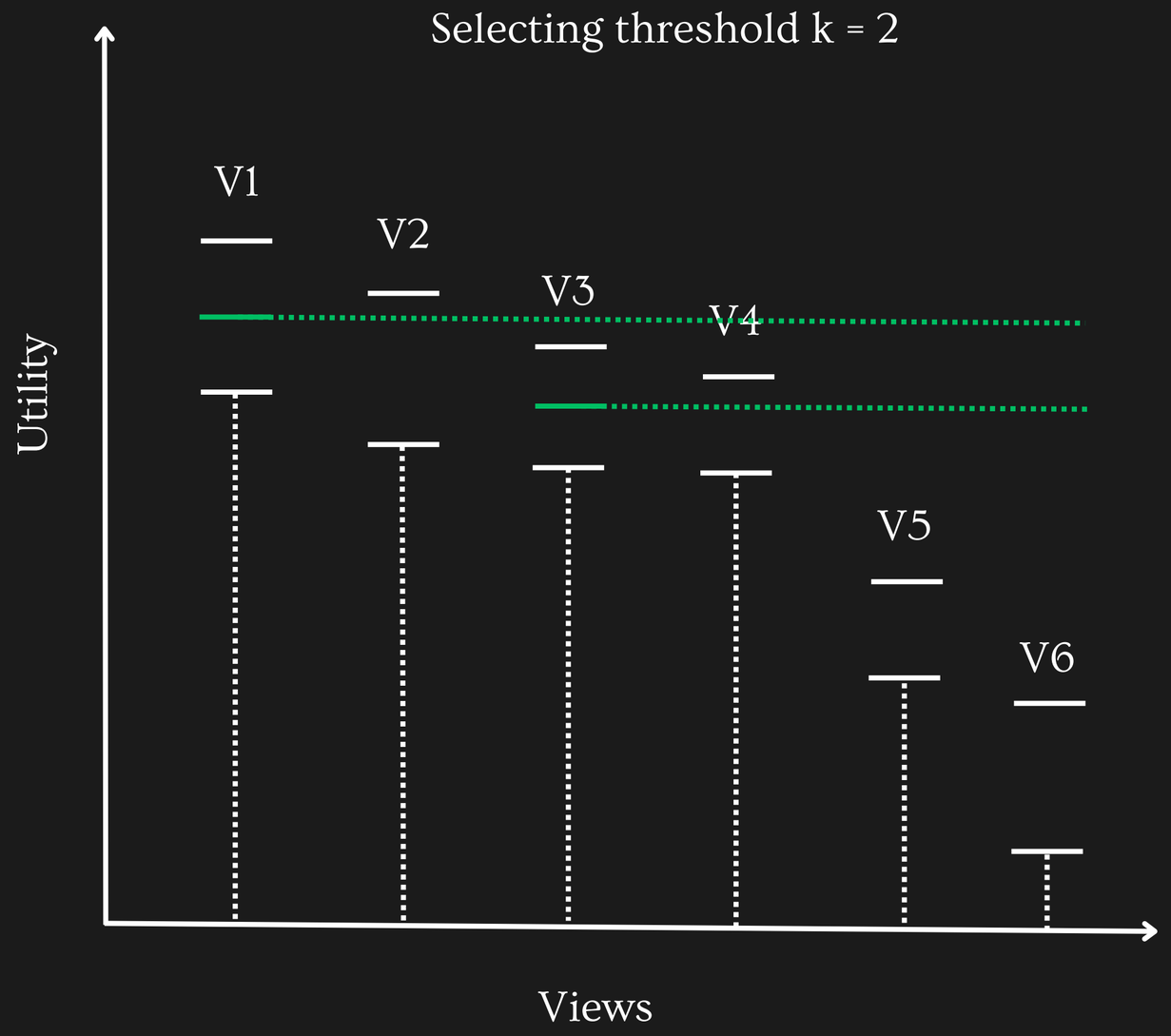
How do you optimize executions by pruning?

Multi-armed bandit pruning



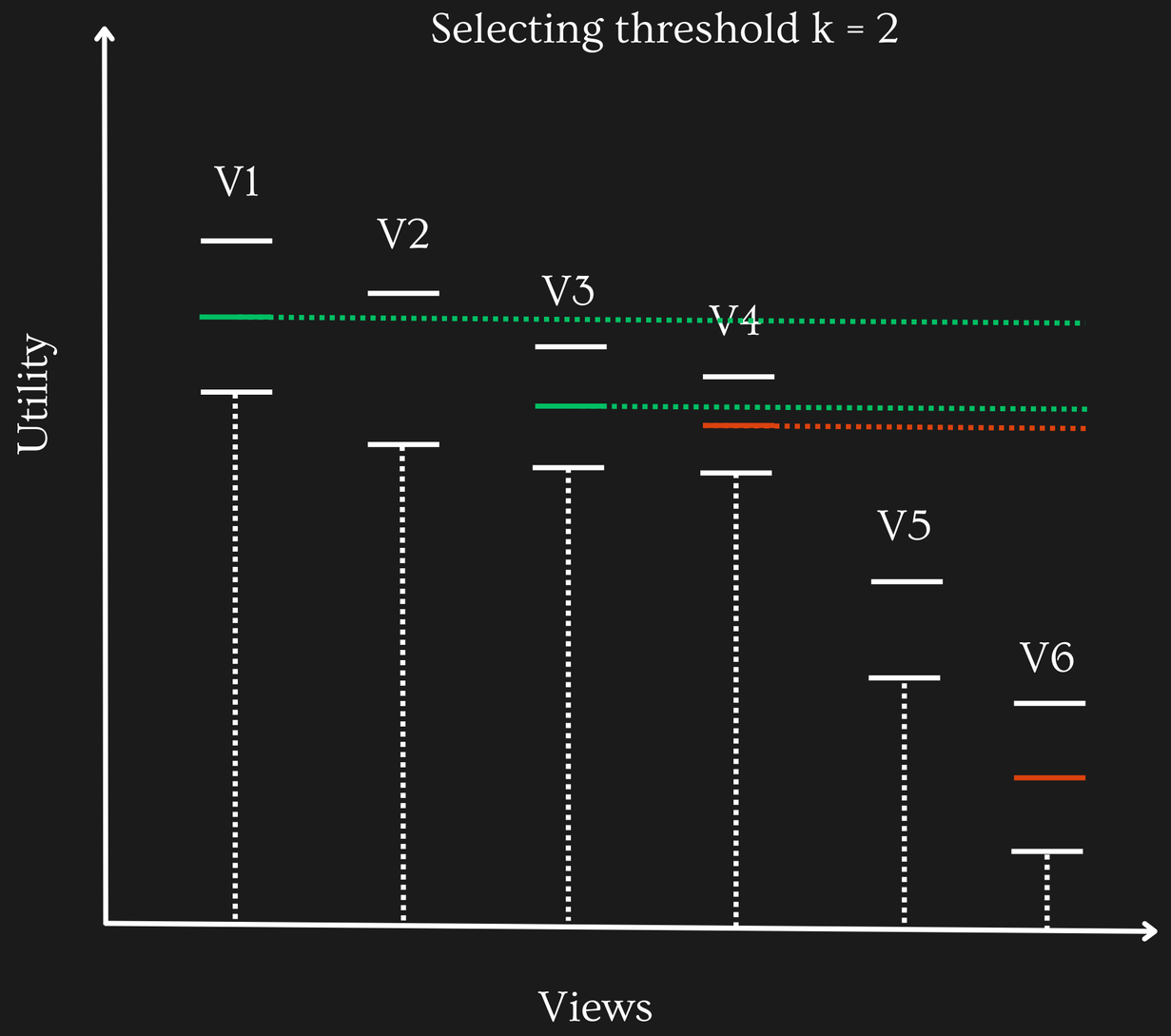
How do you optimize executions by pruning?

Multi-armed bandit pruning



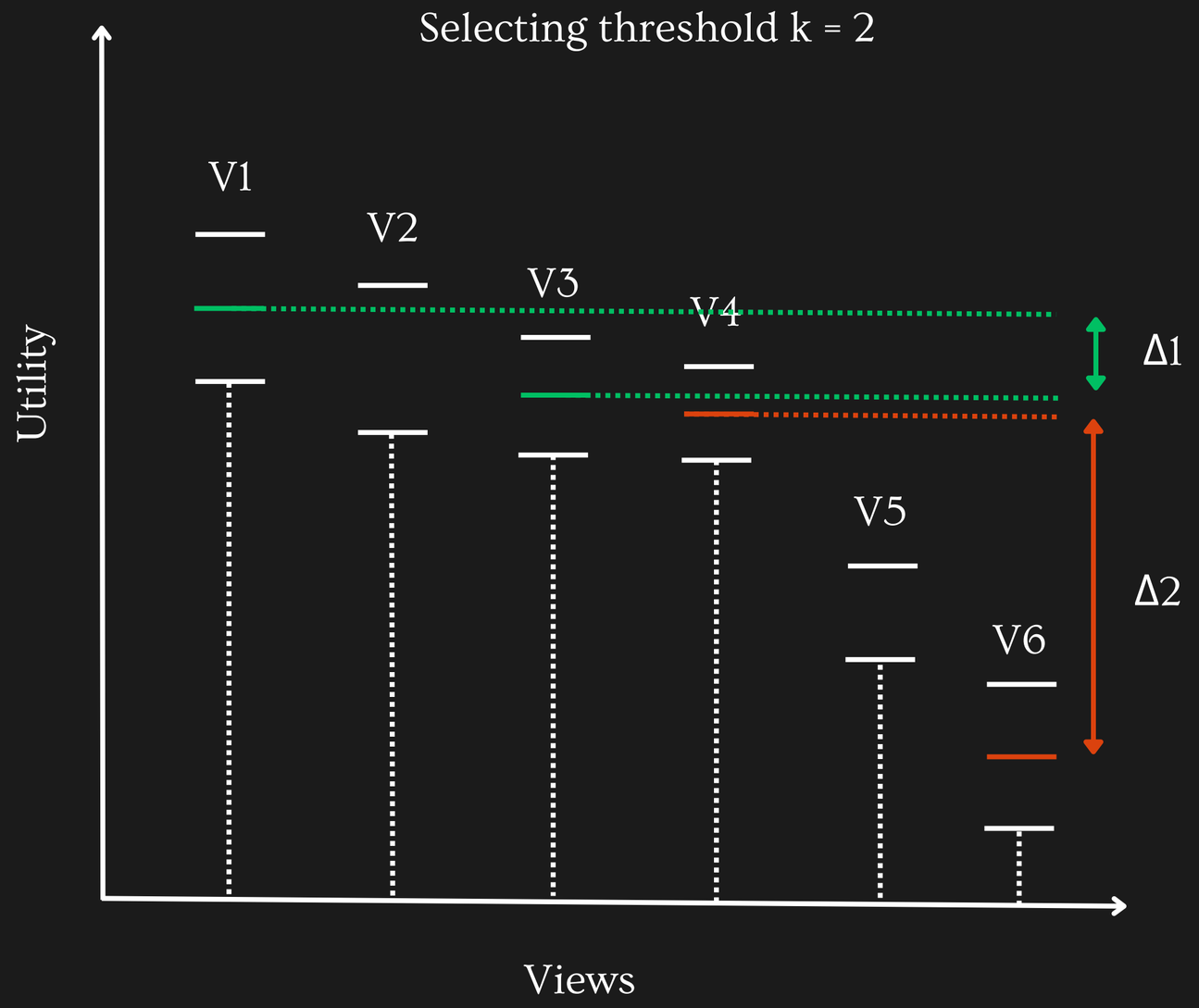
How do you optimize executions by pruning?

Multi-armed bandit pruning



How do you optimize executions by pruning?

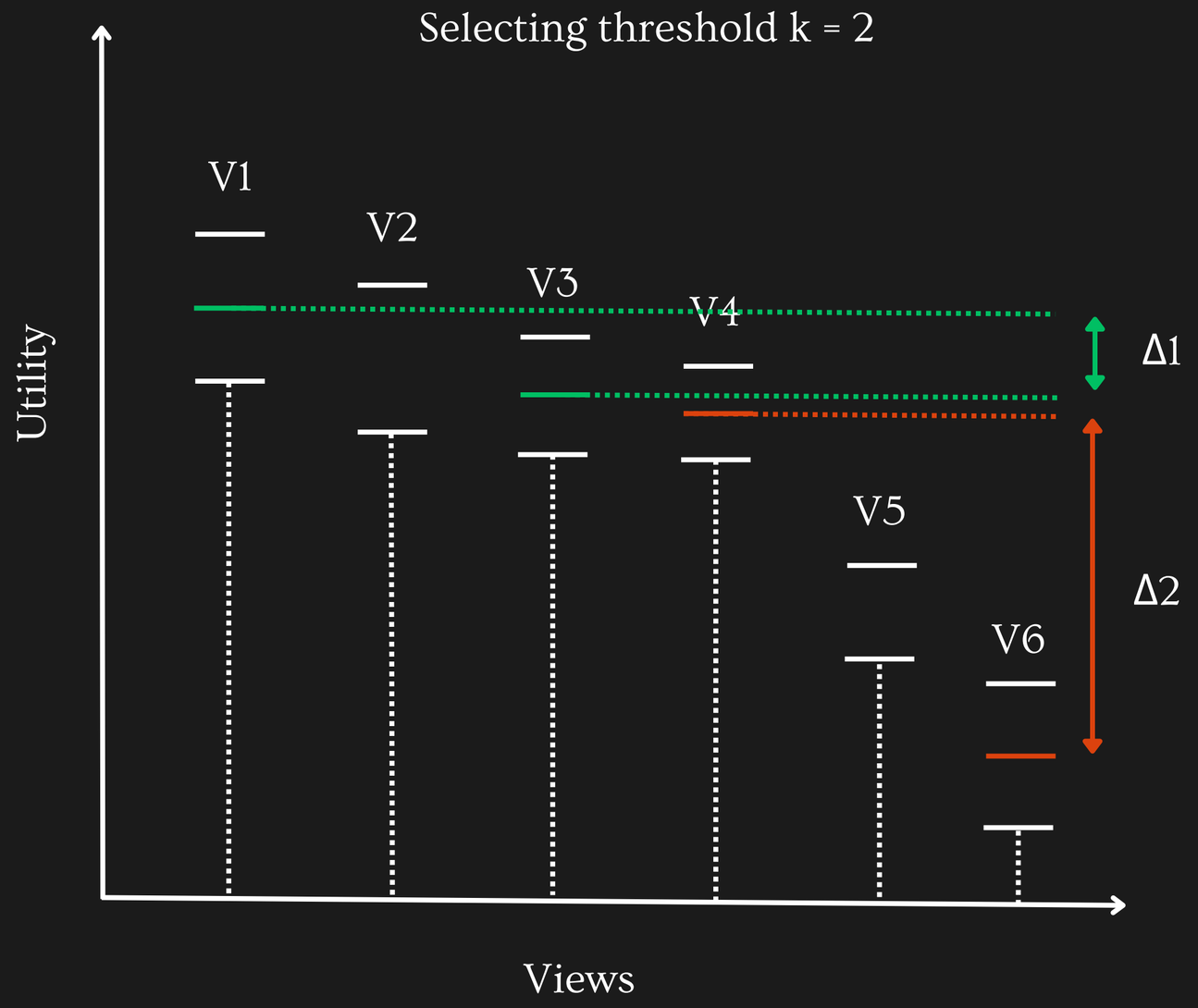
Multi-armed bandit pruning



How do you optimize executions by pruning?

Multi-armed bandit pruning

Δ1 < Δ2, therefore V6 is removed from consideration



How do you optimize executions by pruning?

Confidence interval-based pruning

Multi-armed bandit pruning

We use ‘consistent distance functions’ for distance between distributions, so that -

Pruning results in increasingly better estimates of utility values over time

Evaluation

Key Metric - Latency 

Countermetrics -

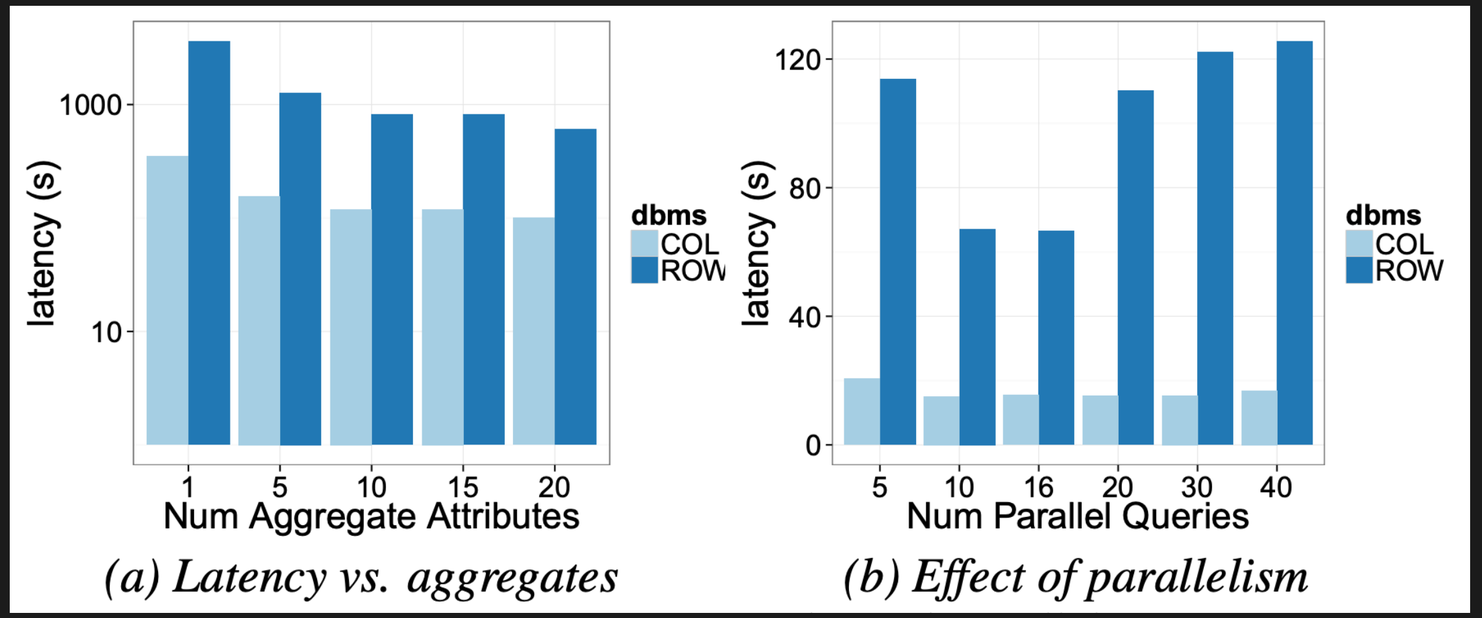
- Accuracy (are the top-k selected views actually the top-k views?)
- Utility distance (are the top-k selected views ‘close’ to the actual top-k views?)

Other considerations -

Scalability (change with size of dataset and # of aggregate views)

Latency Improvement from Sharing

- Combine multiple aggregate view queries
- Combine multiple GROUP BYs
- Combine target and reference queries
- Parallelize query execution



Latency Improvement from Sharing

Combine multiple aggregate view queries

Up to 4x

Combine multiple GROUP BYs

Up to 2.5x

Combine target and reference queries

Up to 2x

Parallelize query execution

Variable*

40x improvement

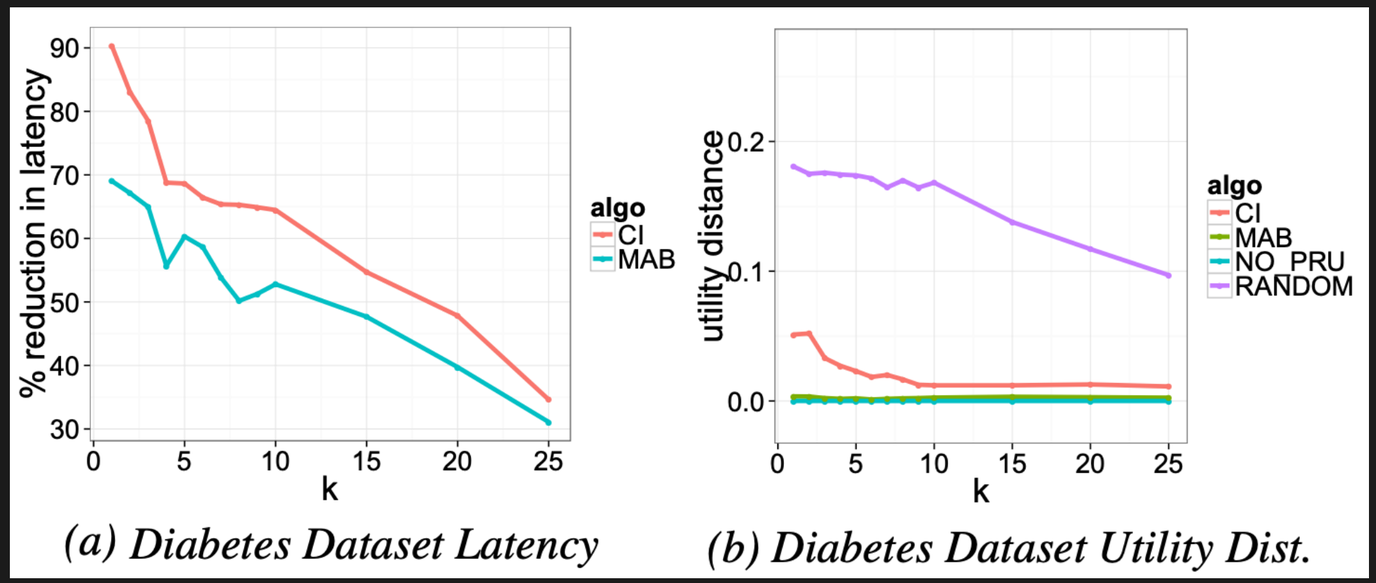
(no effect on accuracy and utility distance)

* depends on system memory and computation constraints

Latency Improvement from Pruning

Confidence interval-based pruning

Multi-armed bandit pruning



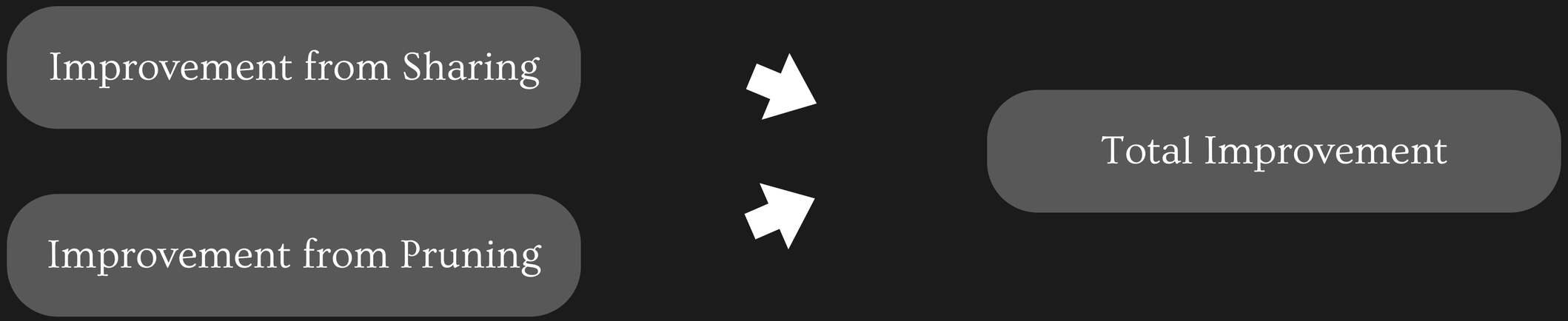
(a) Diabetes Dataset Latency

(b) Diabetes Dataset Utility Dist.

Tradeoff between latency and ‘quality’ -
CI prunes faster, but MAB retains quality (utility distance) better

Latency reduced by over 50% with either technique
For smaller k, latency reduction > 90%

Evaluation



Improvements from each optimization are multiplied

User Study

21

Validate our deviation-based utility metric

Compare SeeDB to a manual charting tool

User Study

5

Validate our deviation-based utility metric

Ground truth

Efficacy of Deviation-based Metric

User Study

5

Validate our deviation-based utility metric

Ground truth

48

4.5

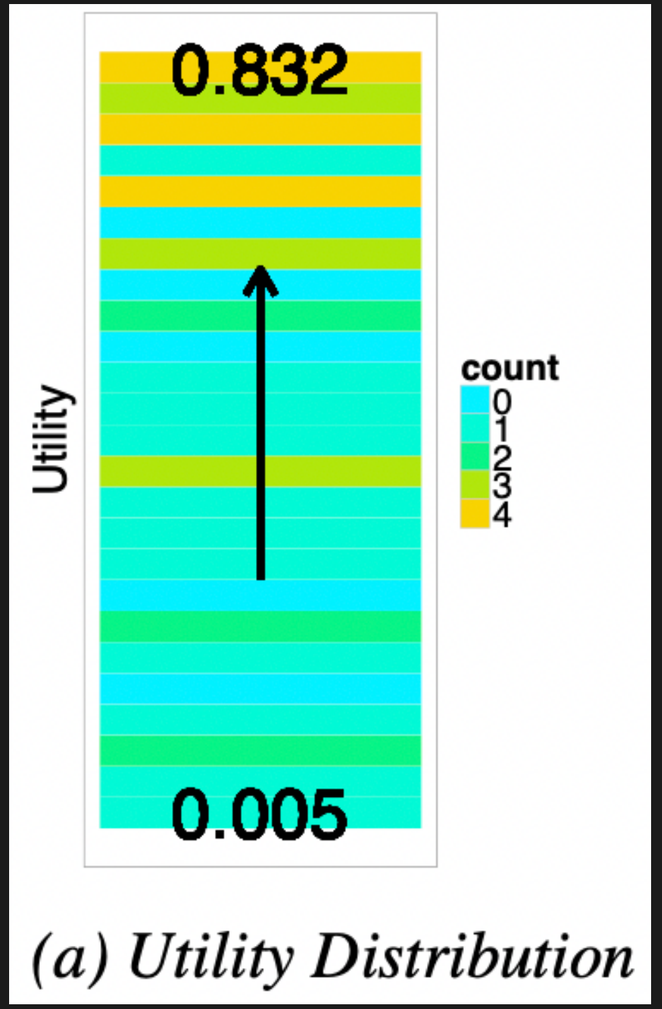
43.5

User Study

5

Validate our deviation-based utility metric

Ground truth



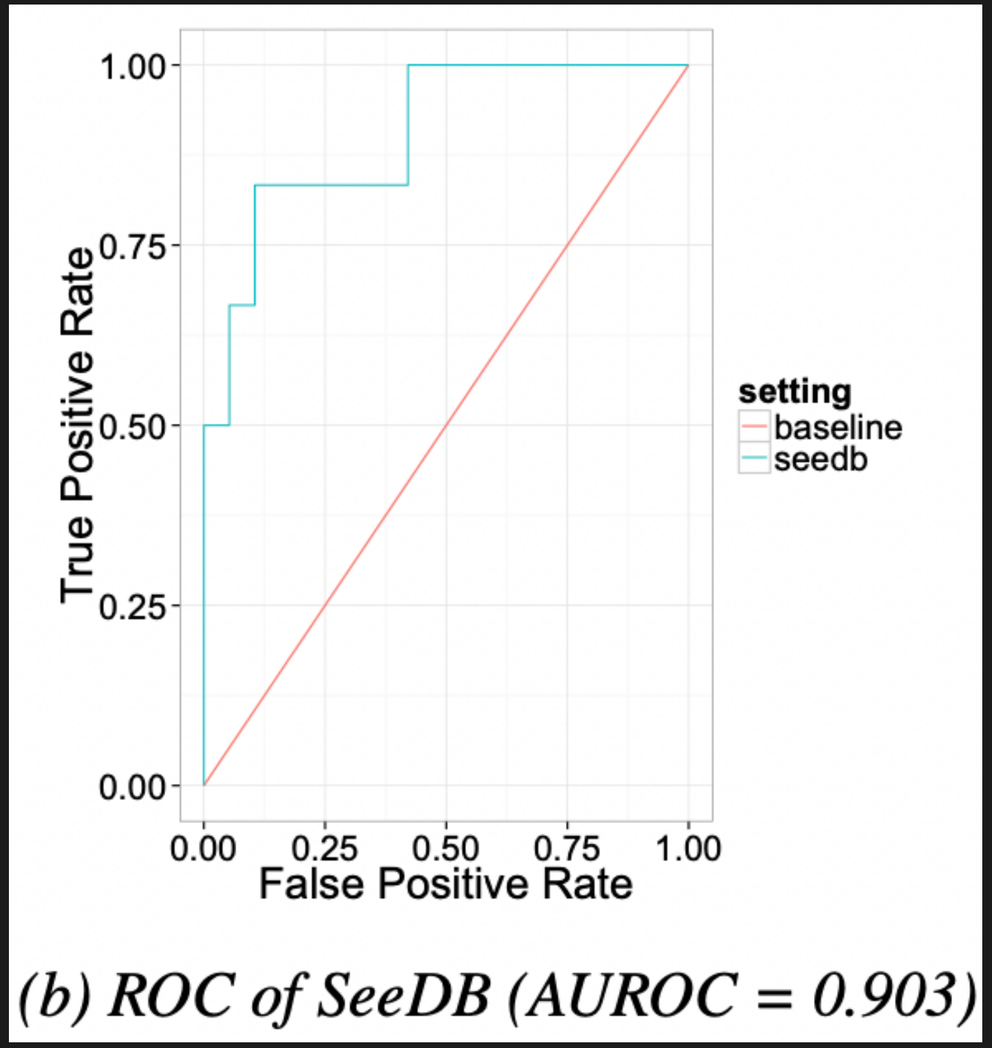
(a) Utility Distribution

User Study

5

Validate our deviation-based utility metric

Efficacy of Deviation-based Metric



(b) ROC of SeeDB (AUROC = 0.903)

User Study

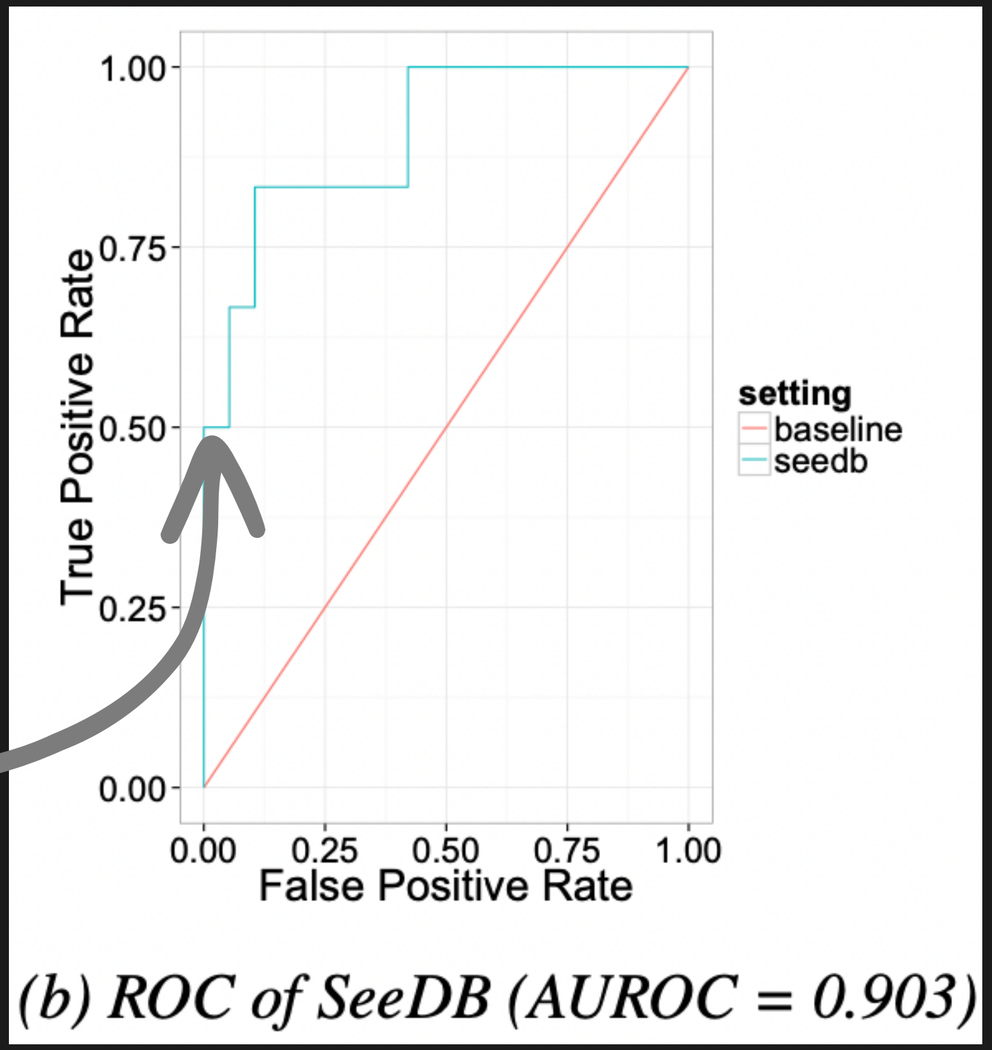
At k = 3

Total Interesting viz = 6

All 3 are interesting

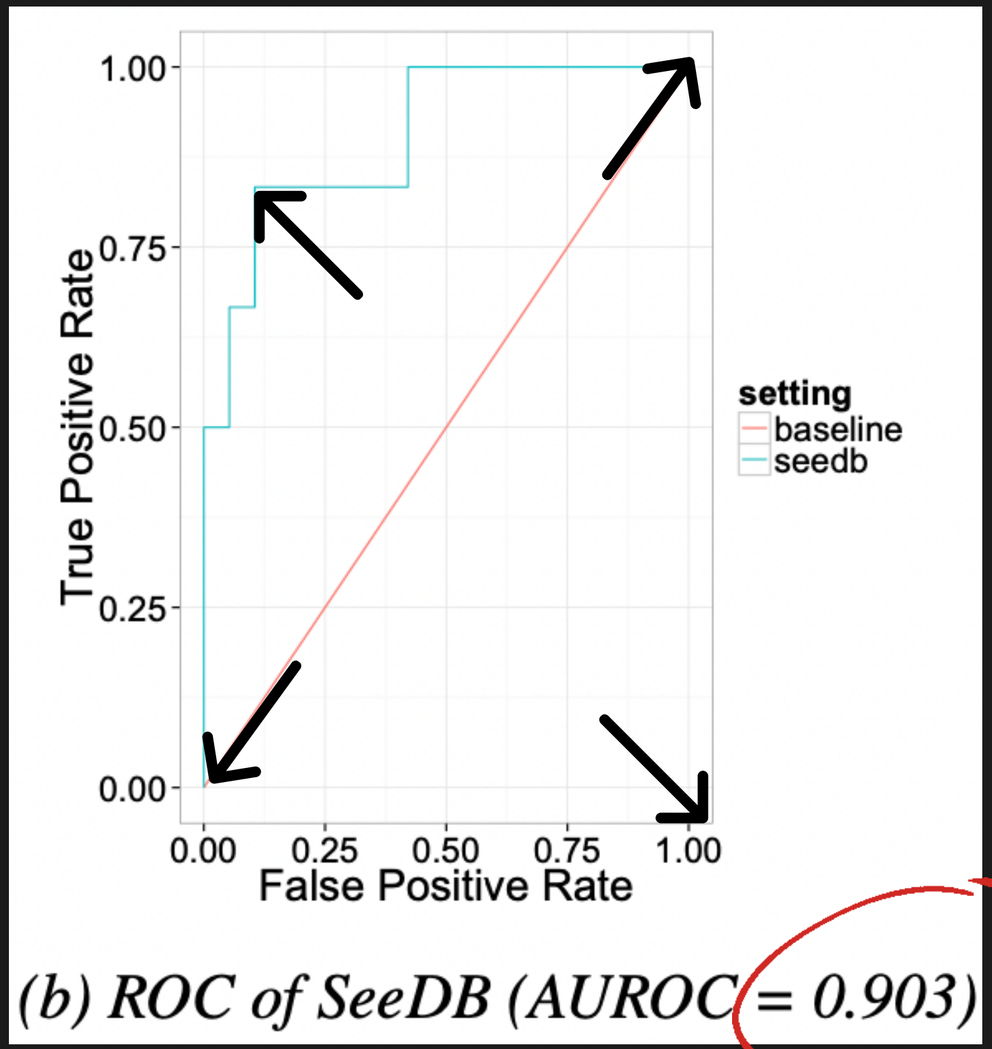
TPR = 3/6 = 0.5

FPR = 0



(b) ROC of SeeDB (AUROC = 0.903)

User Study



(b) ROC of SeeDB (AUROC = 0.903)

User Study

16

SeeDB Manual V/S SeeDB

Compare SeeDB to a manual charting tool

- Interaction logs
- Exit Interview
- Surveys

User Study

16

SeeDB Manual V/S SeeDB

Compare SeeDB to a manual charting tool

Total Visualizations

User Study

16

SeeDB Manual V/S SeeDB

3X

Compare SeeDB to a manual charting tool

Bookmarked Visualizations

User Study

16

SeeDB Manual V/S SeeDB

3X

Compare SeeDB to a manual charting tool

Bookmark Rate

User Study

16

SeeDB Manual V/S SeeDB

Compare SeeDB to a manual charting tool

87% of participants indicated that SeeDB recommendations sped up their visual analysis

All participants preferred SeeDB to Manual

Limitations and Next Steps

- Non-relational databases
- Arbitrary schema
- Real-time data analysis
- Wider variety of visualizations
- Support for other utility functions to enable more high-quality visualizations

Thank you!

AnkitaShanbhag30/SeeDB-Partial-Implementation: Implementation of SeeDB

Implementation of SeeDB. Contribute to AnkitaShanbhag30/SeeDB-Partial-Implementation development by creating an account on GitHub.

GitHub