

# The Price of Tailoring the Index to Your Data:

# **POISONING ATTACKS**

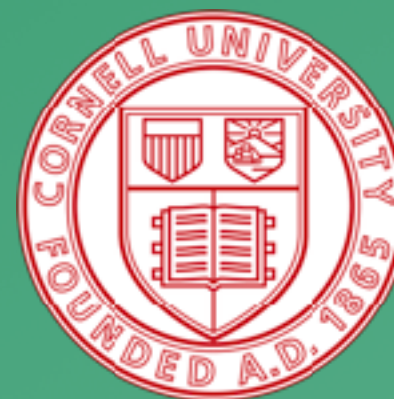
## On Learned Index Structures

Evgenios Kornaropoulos



<https://encrypted.systems>

Silei Ren



Roberto Tamassia

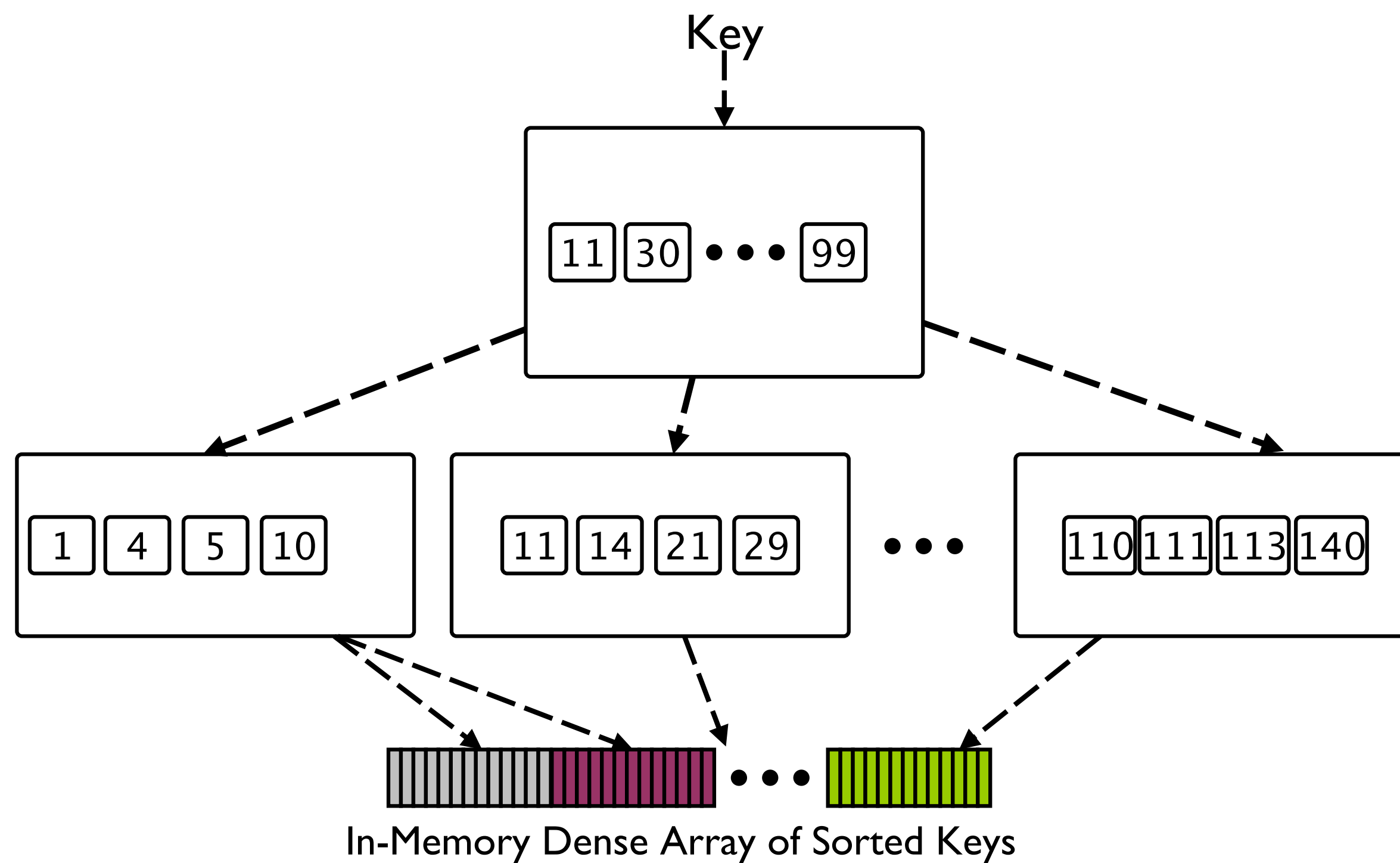




# LEARNED INDEX STRUCTURES

## ML + SYSTEMS

### Classic Algorithmic Approach



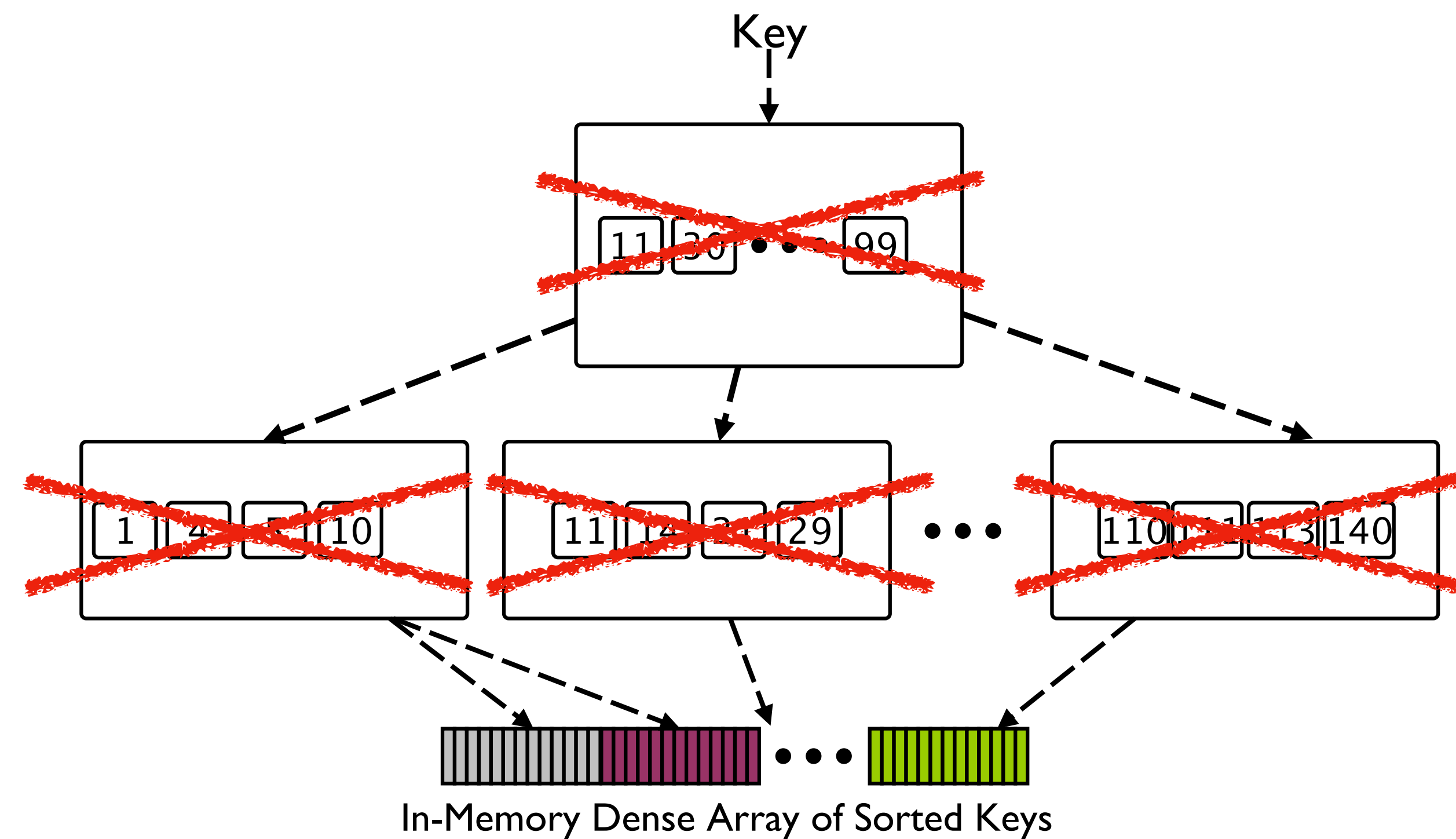
**B-TREE**



# LEARNED INDEX STRUCTURES

## ML + SYSTEMS

~~Classic Algorithmic Approach~~



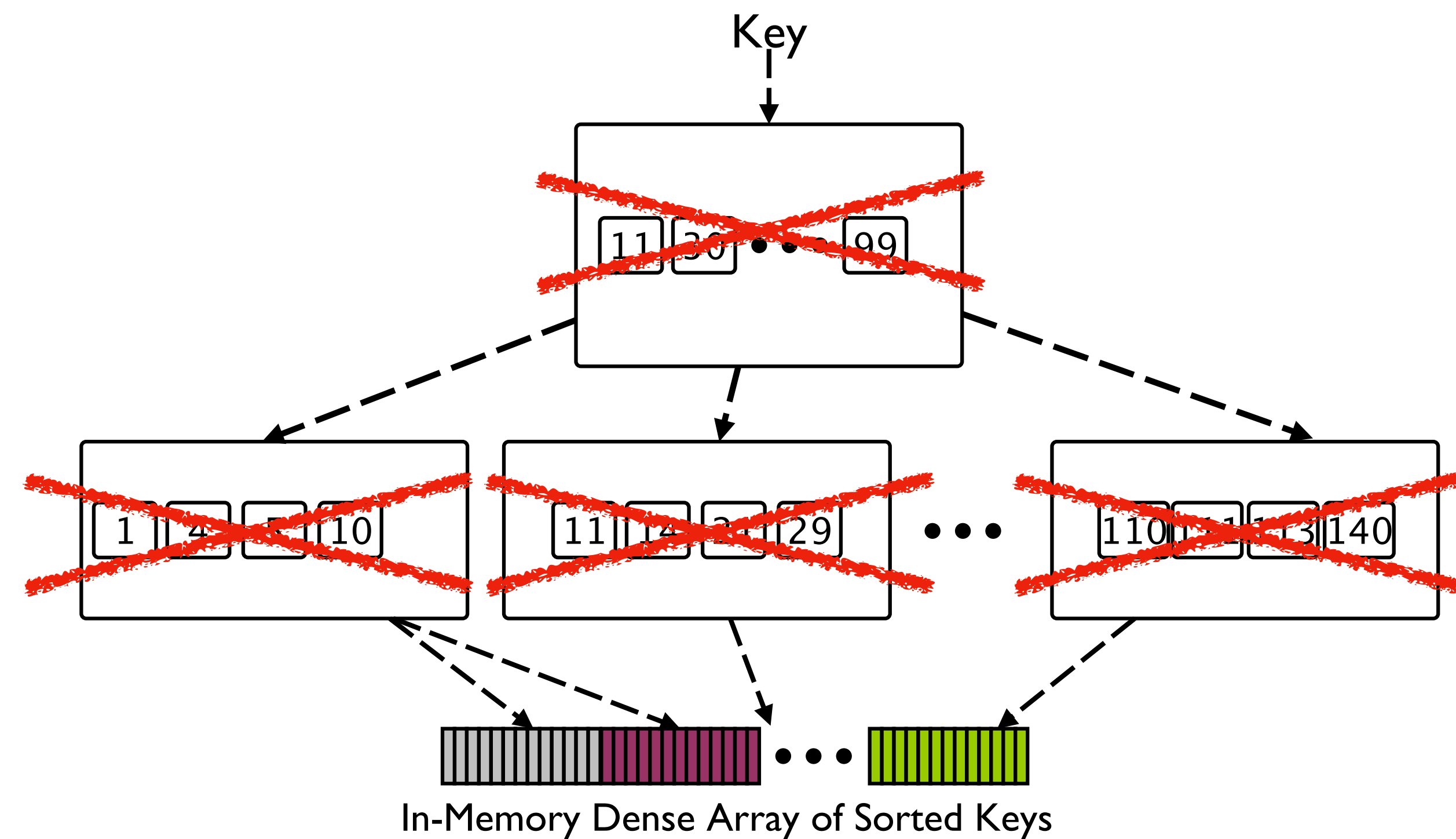
**B-TREE**



# LEARNED INDEX STRUCTURES

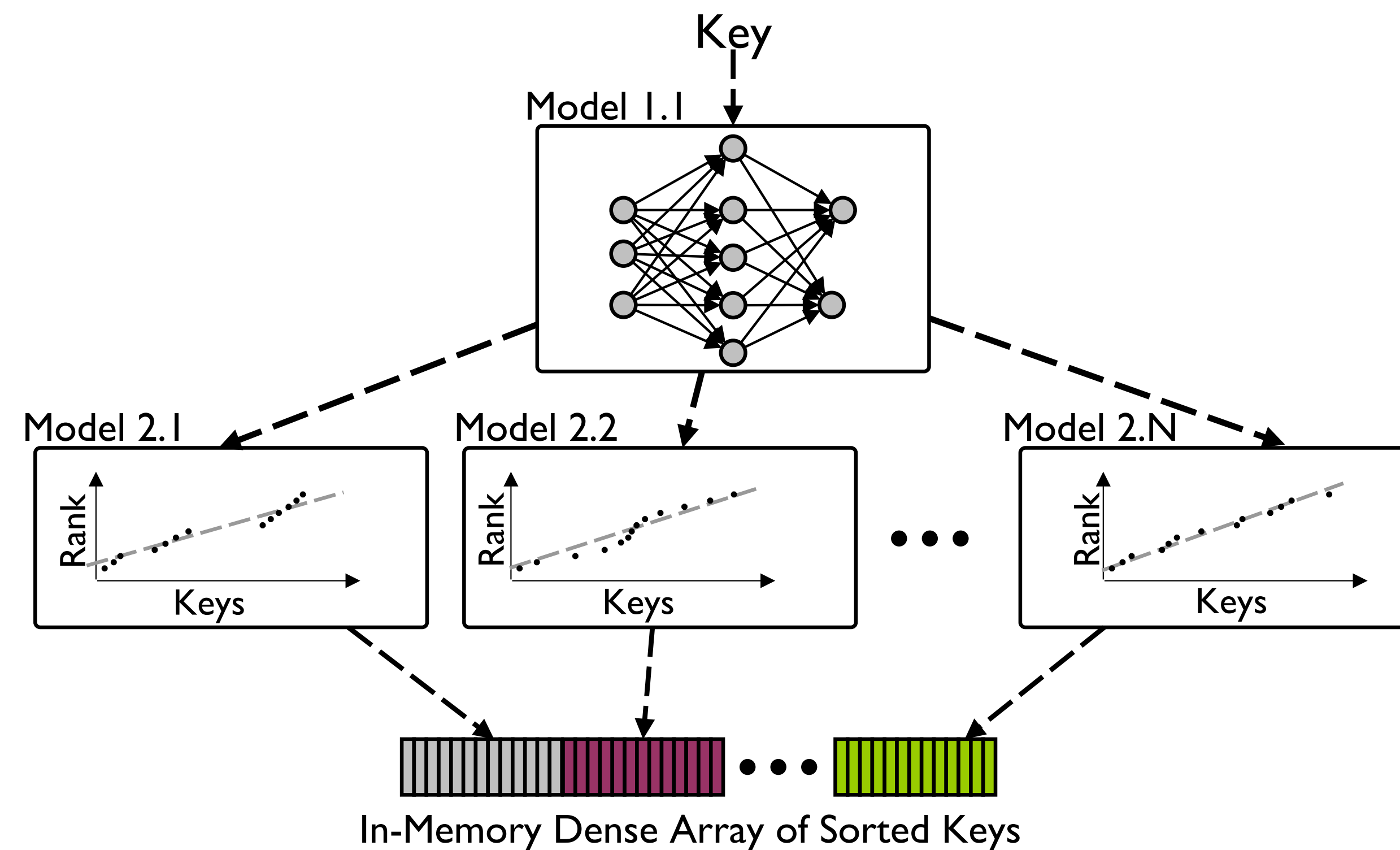
## ML + SYSTEMS

### ~~Classic Algorithmic Approach~~



### B-TREE

### New ML Approach

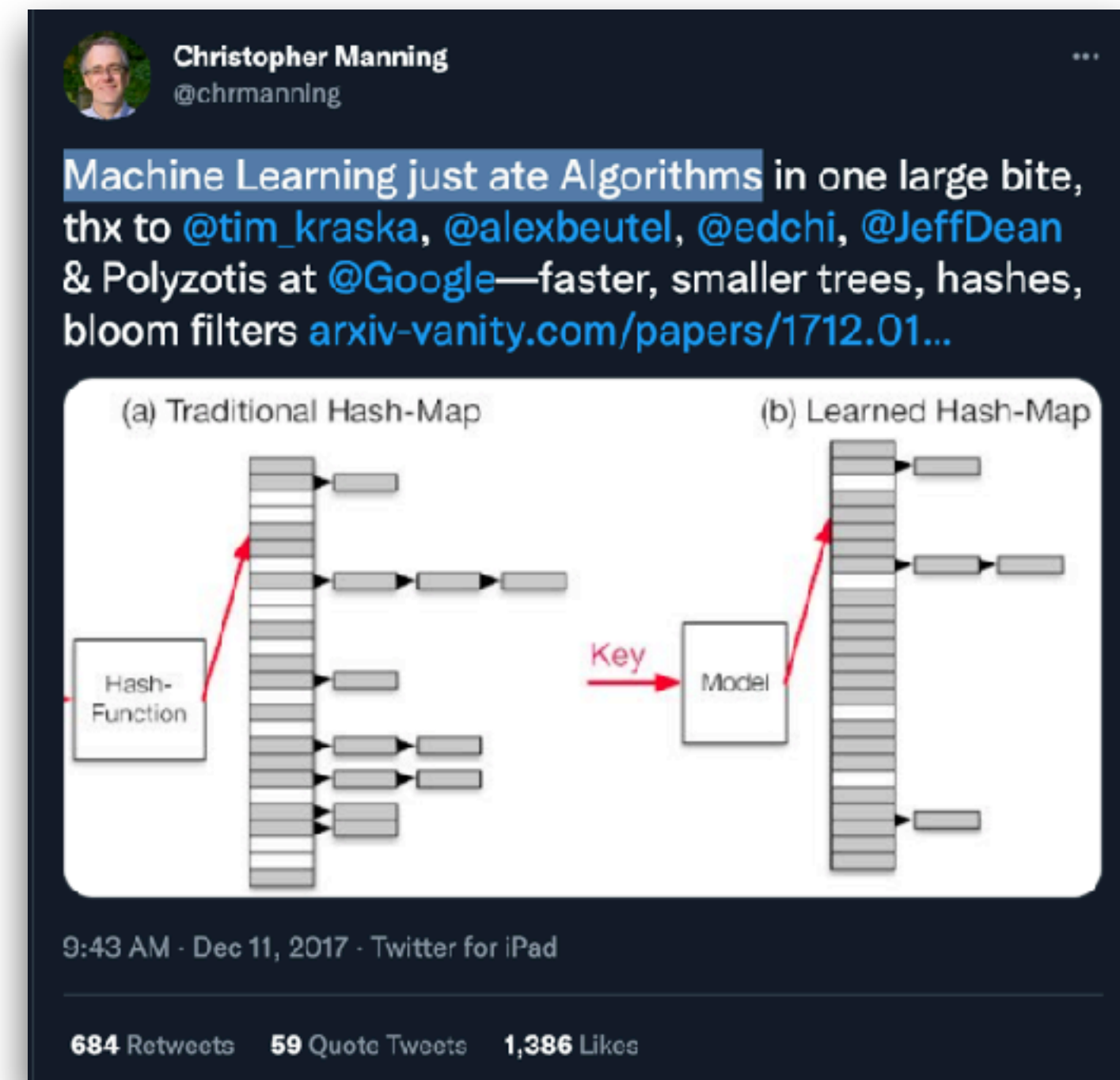
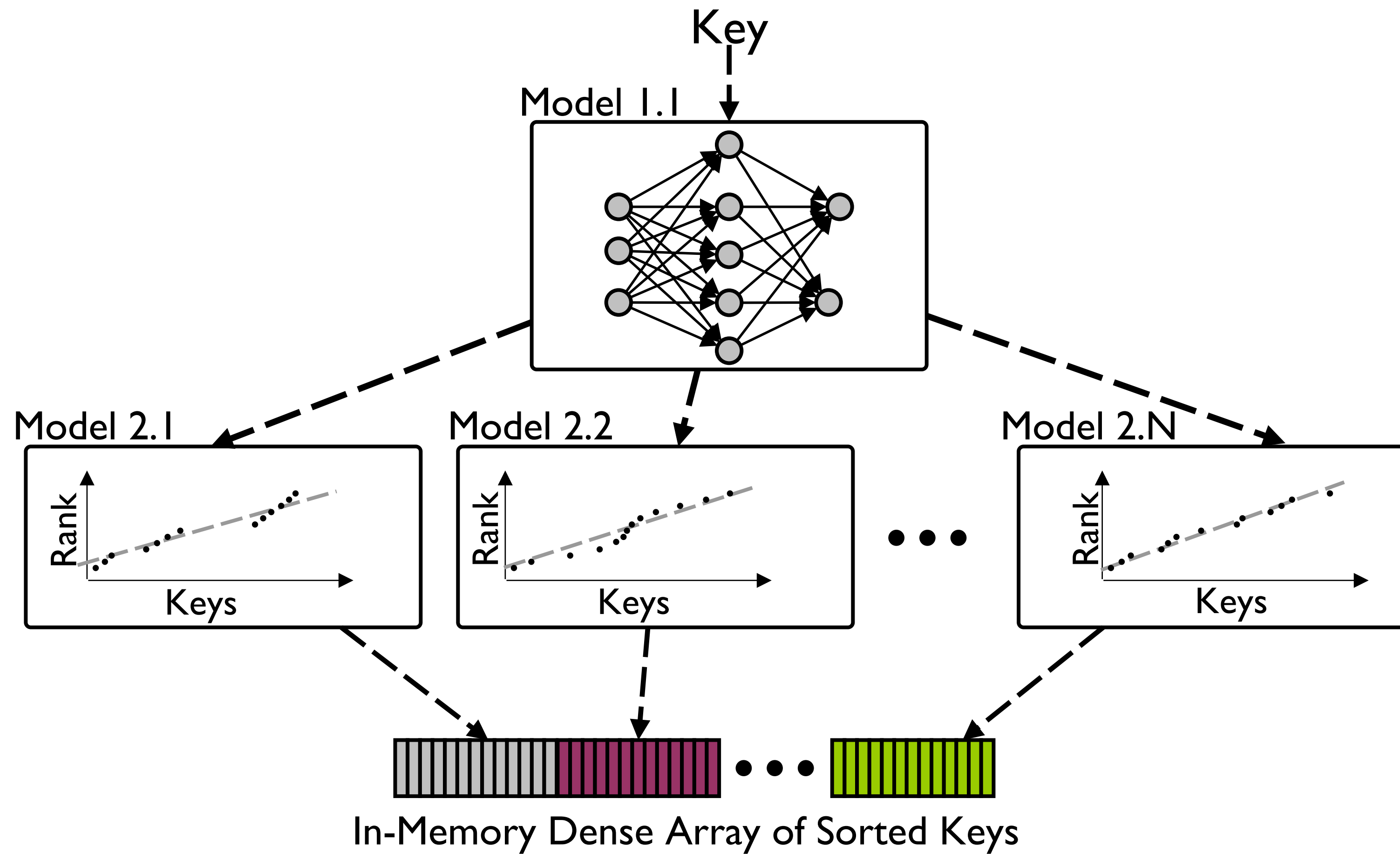


### LEARNED INDEX STRUCTURE (LIS)



# LEARNED INDEX STRUCTURES

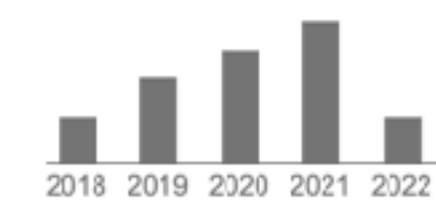
## ML + SYSTEMS



### The case for learned index structures

Authors	Tim Kraska, Alex Beutel, Ed H Chi, Jeffrey Dean, Neoklis Polyzotis
Publication date	2018/5/27
Book	Proceedings of the 2018 international conference on management of data
Pages	489-504
Description	Indexes are models: a B-tree index can be seen as a model to map a key to the position of a record within a sorted array, a Hash index as a model to map a key to a position of a record within an unsorted array, and a BitMap index as a model to indicate if a data record exists or not. In this exploratory research paper, we start from this premise and posit that all existing index structures can be replaced with other types of models, including deep-learning models, which we term learned indexes. We theoretically analyze under which conditions learned indexes outperform traditional index structures and describe the main challenges in designing learned index structures. Our initial results show that our learned indexes can have significant advantages over traditional indexes. More importantly, we believe that the idea of replacing core components of a data management system through learned models has far ...

Total citations Cited by 640





# ML for Systems Papers

This list is incomplete. If we are missing a paper, please email [mlsyspapers@lists.csail.mit.edu](mailto:mlsyspapers@lists.csail.mit.edu) and we will include it. If you would like to be informed about new research papers, subscribe [here](#).

**Acknowledgement:** Parts of this list were sourced from [this repository](#).

## Table of Contents

- [Tutorials / Surveys](#)
- [Learned Range Indexes](#)
- [New Learned Index Applications](#)
- [Learned Multi-Dimensional Indexing & Storage Layouts](#)
- [Learned Bloom Filters](#)
- [Hash Maps / Hashing](#)
- [Partitioning](#)
- [Data Compression](#)
- [Systems and General Optimizations](#)
- [Index Recommendation](#)
- [Configuration Tuning](#)
- [Cardinality / Selectivity Estimation](#)
- [Data-based Cardinality Estimation](#)
- [Query-based Cardinality Estimation](#)
- [Cost Estimation](#)
- [Query Optimization](#)
- [Query Processing](#)
- [Scheduling](#)
- [Caching](#)
- [Sorting](#)
- [Garbage Collection](#)
- [Sketches](#)
- [Compilation / Compilers](#)
- [SQL-Related](#)
- [Workload Related](#)
- [Data Cleaning and Exploration](#)



# ML for Systems Papers

This list is incomplete. If we are missing a paper, please email [mlsyspapers@lists.csail.mit.edu](mailto:mlsyspapers@lists.csail.mit.edu) and we will include it. If you would like to be informed about new research papers, subscribe [here](#).

**Acknowledgement:** Parts of this list were sourced from [this repository](#).

## Table of Contents

- [Tutorials / Surveys](#)
- [Learned Range Indexes](#)
- [New Learned Index Applications](#)
- [Learned Multi-Dimensional Indexing & Storage Layouts](#)
- [Learned Bloom Filters](#)
- [Hash Maps / Hashing](#)
- [Partitioning](#)
- [Data Compression](#)
- [Systems and General Optimizations](#)
- [Index Recommendation](#)
- [Configuration Tuning](#)
- [Cardinality / Selectivity Estimation](#)
- [Data-based Cardinality Estimation](#)
- [Query-based Cardinality Estimation](#)
- [Cost Estimation](#)
- [Query Optimization](#)
- [Query Processing](#)
- [Scheduling](#)
- [Caching](#)
- [Sorting](#)
- [Garbage Collection](#)
- [Sketches](#)
- [Compilation / Compilers](#)
- [SQL-Related](#)
- [Workload Related](#)
- [Data Cleaning and Exploration](#)

285 papers

What is the Price of  
**Learning**  
the Patterns in the Data?



# THIS WORK

## ML ATTACKS ON LEARNED INDEX STRUCTURES

### The Price of Tailoring the Index to Your Data: Poisoning Attacks on Learned Index Structures

Evgenios M. Kornaropoulos  
George Mason University, USA  
evgenios@gmu.edu

Silei Ren  
Cornell University, USA  
sr2262@cornell.edu

Roberto Tamassia  
Brown University, USA  
roberto@tamassia.net

#### ABSTRACT

The concept of *learned index structures* relies on the idea that the input-output functionality of a database index can be viewed as a prediction task and, thus, implemented using a machine learning model instead of traditional algorithmic techniques. This novel angle for a decades-old problem has inspired exciting results at the intersection of machine learning and data structures. However, the advantage of learned index structures, i.e., the ability to adjust to the data at hand via the underlying ML model, can become a disadvantage from a security perspective as it could be exploited.

In this work, we present the first study of data poisoning attacks on learned index structures. Our poisoning approach is different from all previous works since the model under attack is trained on a cumulative distribution function (CDF) and, thus, every injection on the training set has a cascading impact on multiple data values. We formulate the first poisoning attacks on linear regression models trained on a CDF, which is a basic building block of the proposed learned index structures. We generalize our poisoning techniques to attack the advanced two-stage design of learned index structures called recursive model index (RMI), which has been shown to outperform traditional B-Trees. We evaluate our attacks under a variety of parametrizations of the model and show that the error of the RMI increases up to 340x and the error of its second-stage models increases up to 1000x.

#### CCS CONCEPTS

• Information systems → Data structures; • Security and privacy → Cryptanalysis and other attacks; • Computing methodologies → Machine learning approaches.

#### KEYWORDS

Learned Systems, Data Poisoning, Attacks, Indexing

#### 1 INTRODUCTION

Database systems rely on *index structures* to access stored data efficiently. It is known to the database community that the motto “one size fits all” does not apply to traditional indexing schemes [24] since each index provides different performance guarantees that depend on the access pattern, the nature of the workload, and the underlying hardware. Even after choosing an appropriate index structure for a specific application, it is usually the case that a database administrator has to manually fine-tune the parameters of the system, either through experience or with help from tools. The work by Kraska, Beutel, Chi, Dea, and Polyzotis [30] challenged the state of affairs by re-framing index structures as a *machine learning* problem where the index directs a query to a memory location(s) based on a trained model tailored on the data at hand.

**Learned Index Structures.** The core idea of a *learned index structure* (LIS) is to model a data structure as a *prediction task*, i.e., get an input key and predict its location in a sorted sequence of key-record pairs. This approach allows the use of (i) continuous functions to encode the data, and (ii) *learning algorithms* to approximate the function. The specific LIS approach proposed by Kraska *et al.* [30] is to build the *cumulative distribution function* (CDF) for the keys. Given a key,  $k$ , the CDF returns the probability that a key chosen according to this distribution takes value less than or equal to  $k$ . Since the above probability is built from the set of keys at hand, it is expressed as the ratio of the number of keys less than  $k$  to the total number of keys. Given this insight, one can use the CDF to (i) compute the number of keys less than the (queried) key  $k$ , and (ii) infer the key’s memory location assuming the keys were sorted during the initialization. Therefore, a simple linear regression on the CDF gives an approximate location of the queried key. Indeed a linear regression on the CDF is one of the building blocks that has been shown to work well [30] and can be combined with *hierarchical models*, also called recursive model index (RMI) structures, so as to balance the final model for latency, memory usage, and computational cost. The hierarchy can be seen as building a mixture of “experts” [39] responsible for subsets of the data. The notion of a LIS has spurred a surge of works that blend ideas from machine learning, data structures, and systems (e.g., [5, 7, 10, 12–14, 17–21, 23, 24, 29, 31, 35, 36, 41, 43, 45, 47–49, 52, 55, 56, 58, 59]).

**First Vulnerability Assessment of Learned Index.** As promising as it may sound to combine ideas from machine learning and data structures, no analysis has been performed to understand potential vulnerabilities of the LIS paradigm. Intuitively, the advantage of a LIS is that the model adapts to the data at hand. However this efficiency might be problematic if the adversary is capable of injecting *maliciously crafted data* before the training of the model, i.e., at the initialization stage of the index structure, so as to cause inaccurate predictions of the location of legitimate data.

The technique of *data poisoning* has been known to be an effective attack vector for over a decade, e.g., see the references in [26]. In the context of static index structures, we focus on the case where the data stored in the index comes from multiple sources as different entities directly or indirectly contribute data, e.g., by generating data with their actions or behavior. A malicious actor can tailor its contributed data to deteriorate the index performance. Indeed, the real-world datasets used in the original LIS work [30] come from multiple contributors and thus, are susceptible to poisoning attacks. Other examples of indexed data generated by multiple sources include data from personalized medicine, where patients voluntarily contribute their own data, as well as cybersecurity analytics where any user can submit its own indicators of compromise. Our threat model, much like all poisoning works [3, 4, 26, 60, 51], assumes that

- New Poisoning Attacks on Cumulative Distribution Functions (CDF)
- Apply Attacks on Hierarchical Learned Indexes
- Test Attack on the Same Datasets + Measure Error due to Poisoning



# THIS WORK

## ML ATTACKS ON LEARNED INDEX STRUCTURES

### The Price of Tailoring the Index to Your Data: Poisoning Attacks on Learned Index Structures

Evgerios M. Kornaropoulos  
George Mason University, USA  
evgenios@gmu.edu

Silei Ren  
Cornell University, USA  
sr2262@cornell.edu

Roberto Tamassia  
Brown University, USA  
roberto@tamassia.net

#### ABSTRACT

The concept of *learned index structures* relies on the idea that the input-output functionality of a database index can be viewed as a prediction task and, thus, implemented using a machine learning model instead of traditional algorithmic techniques. This novel angle for a decades-old problem has inspired exciting results at the intersection of machine learning and data structures. However, the advantage of learned index structures, i.e., the ability to adjust to the data at hand via the underlying ML model, can become a disadvantage from a security perspective as it could be exploited.

In this work, we present the first study of data poisoning attacks on learned index structures. Our poisoning approach is different from all previous works since the model under attack is trained on a cumulative distribution function (CDF) and, thus, every injection on the training set has a cascading impact on multiple data values. We formulate the first poisoning attacks on linear regression models trained on a CDF, which is a basic building block of the proposed learned index structures. We generalize our poisoning techniques to attack the advanced two-stage design of learned index structures called recursive model index (RMI), which has been shown to outperform traditional B-Trees. We evaluate our attacks under a variety of parametrizations of the model and show that the error of the RMI increases up to 340x and the error of its second-stage models increases up to 1000x.

#### CCS CONCEPTS

• Information systems → Data structures; • Security and privacy → Cryptanalysis and other attacks; • Computing methodologies → Machine learning approaches.

#### KEYWORDS

Learned Systems, Data Poisoning, Attacks, Indexing

#### 1 INTRODUCTION

Database systems rely on *index structures* to access stored data efficiently. It is known to the database community that the motto

**Learned Index Structures.** The core idea of a *learned index structure* (LIS) is to model a data structure as a *prediction task*, i.e., get an input key and predict its location in a sorted sequence of key-record pairs. This approach allows the use of (i) continuous functions to encode the data, and (ii) *learning algorithms* to approximate the function. The specific LIS approach proposed by Krasla *et al.* [36] is to build the *cumulative distribution function* (CDF) for the keys. Given a key,  $k$ , the CDF returns the probability that a key chosen according to this distribution takes value less than or equal to  $k$ . Since the above probability is built from the set of keys at hand, it is expressed as the ratio of the number of keys less than  $k$  to the total number of keys. Given this insight, one can use the CDF to (i) compute the number of keys less than the (queried) key  $k$ , and (ii) infer the key's memory location assuming the keys were sorted during the initialization. Therefore, a simple linear regression on the CDF gives an approximate location of the queried key. Indeed a linear regression on the CDF is one of the building blocks that has been shown to work well [30] and can be combined with *hierarchical models*, also called recursive model index (RMI) structures, so as to balance the final model for latency, memory usage, and computational cost. The hierarchy can be seen as building a mixture of "experts" [39] responsible for subsets of the data. The notion of a LIS has spurred a surge of works that blend ideas from machine learning, data structures, and systems (e.g., [5, 7, 10, 12–14, 17–21, 23, 24, 29, 31, 35, 36, 41, 43, 45, 47–49, 52, 55, 56, 58, 59]).

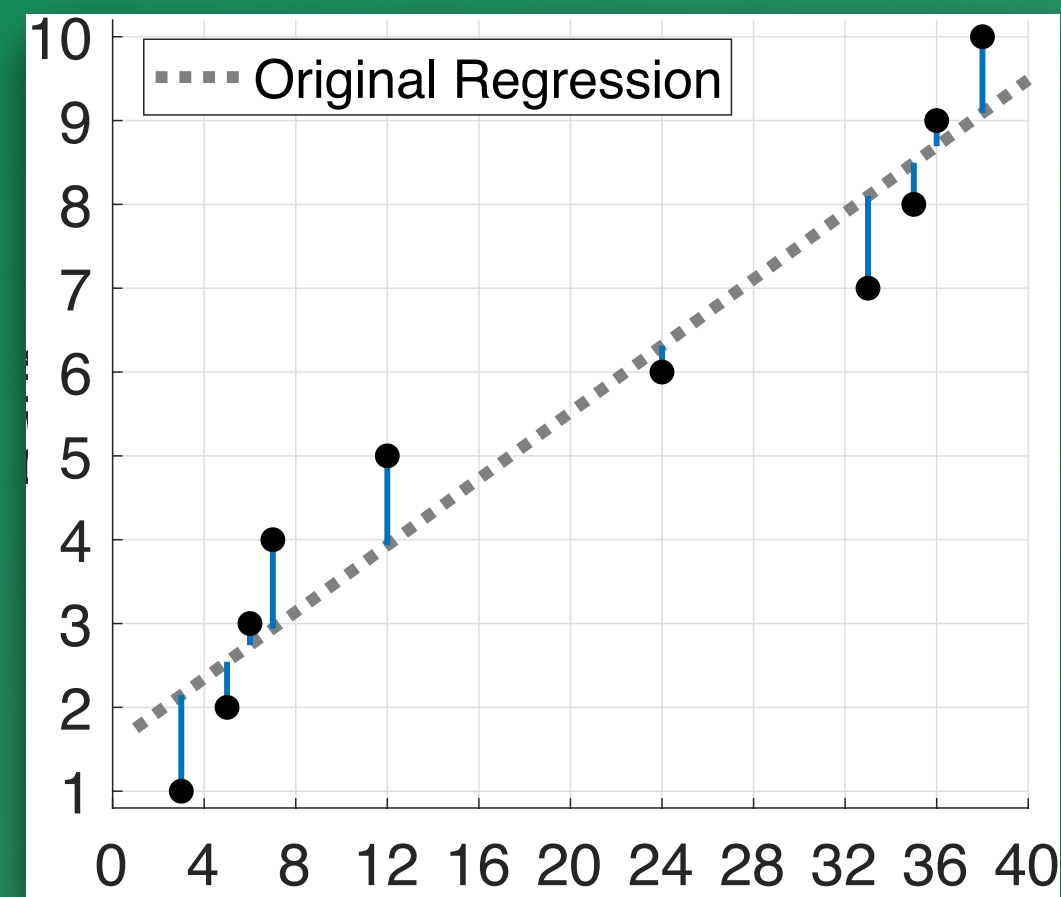
**First Vulnerability Assessment of Learned Index.** As promising as it may sound to combine ideas from machine learning and data structures, no analysis has been performed to understand potential vulnerabilities of the LIS paradigm. Intuitively, the advantage of a LIS is that the model adapts to the data at hand. However this efficiency might be problematic if the adversary is capable of injecting *maliciously crafted data* before the training of the model, i.e., at the initialization stage of the index structure, so as to cause inaccurate predictions of the location of legitimate data.

The technique of *data poisoning* has been known to be an effective attack vector for over a decade, e.g., see the references in [26]. In the context of static index structures, we focus on the case where the data stored in the index comes from multiple sources as different

- **New Poisoning Attacks** on Cumulative Distribution Functions (CDF)
- Apply Attacks on **Hierarchical** Learned Indexes
- Test Attack on the Same Datasets + Measure Error due to Poisoning

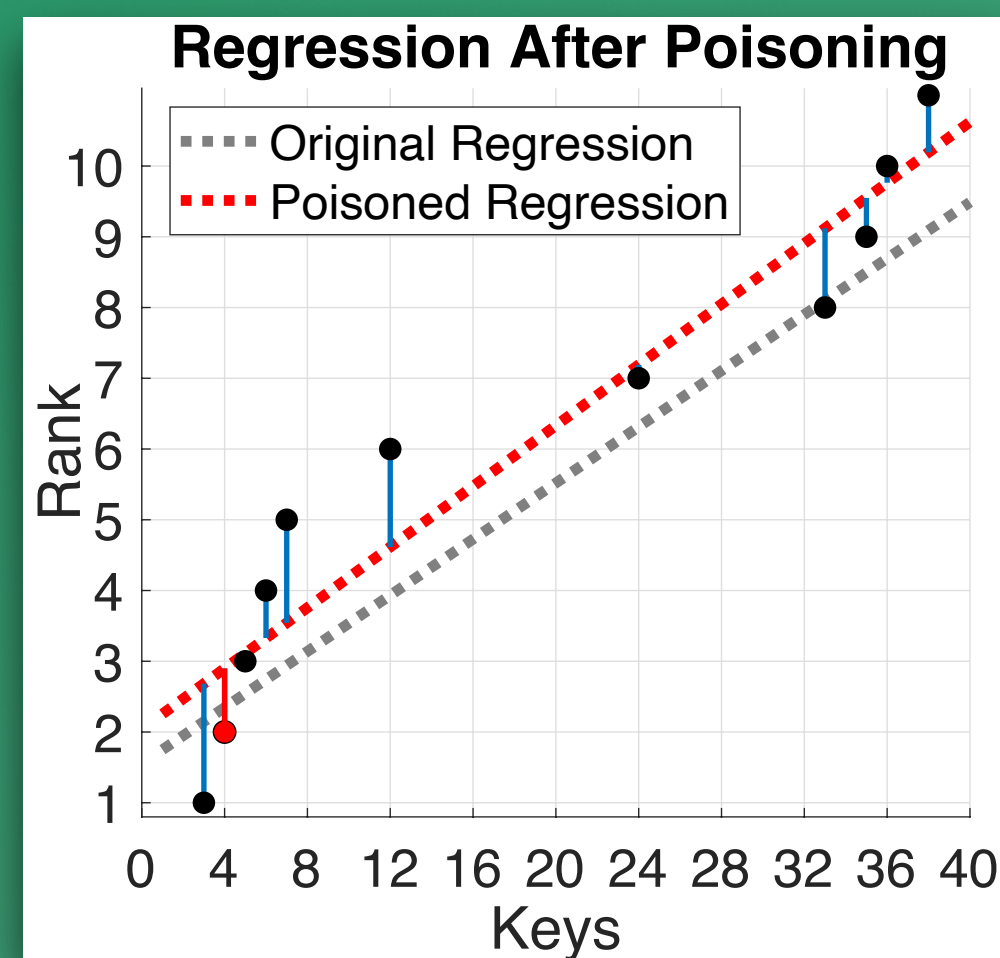
**Need to understand the worst-case behavior of learned models on data**

# Part-1



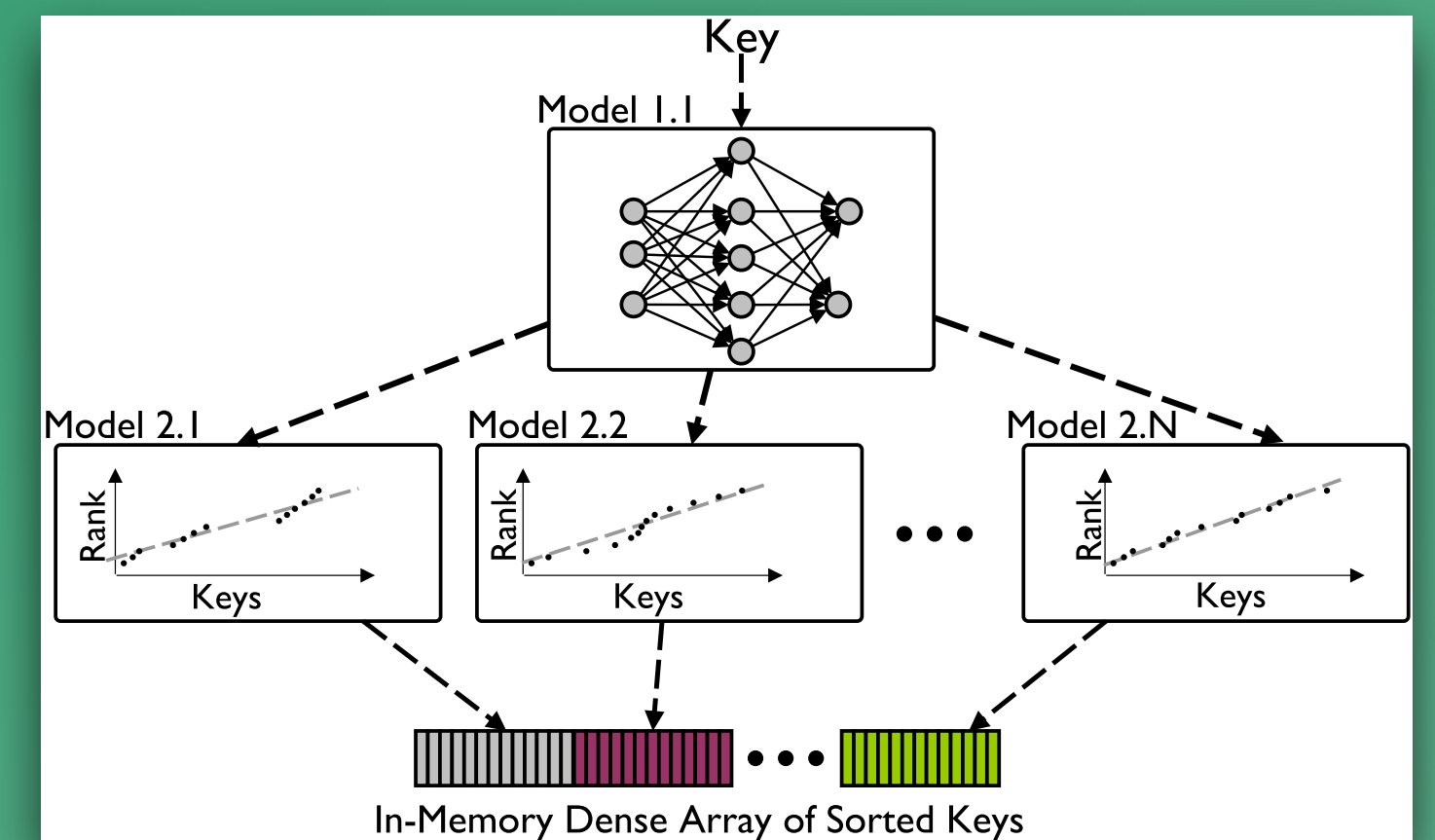
Attacks on  
(Vanilla) Regression

# Part-2



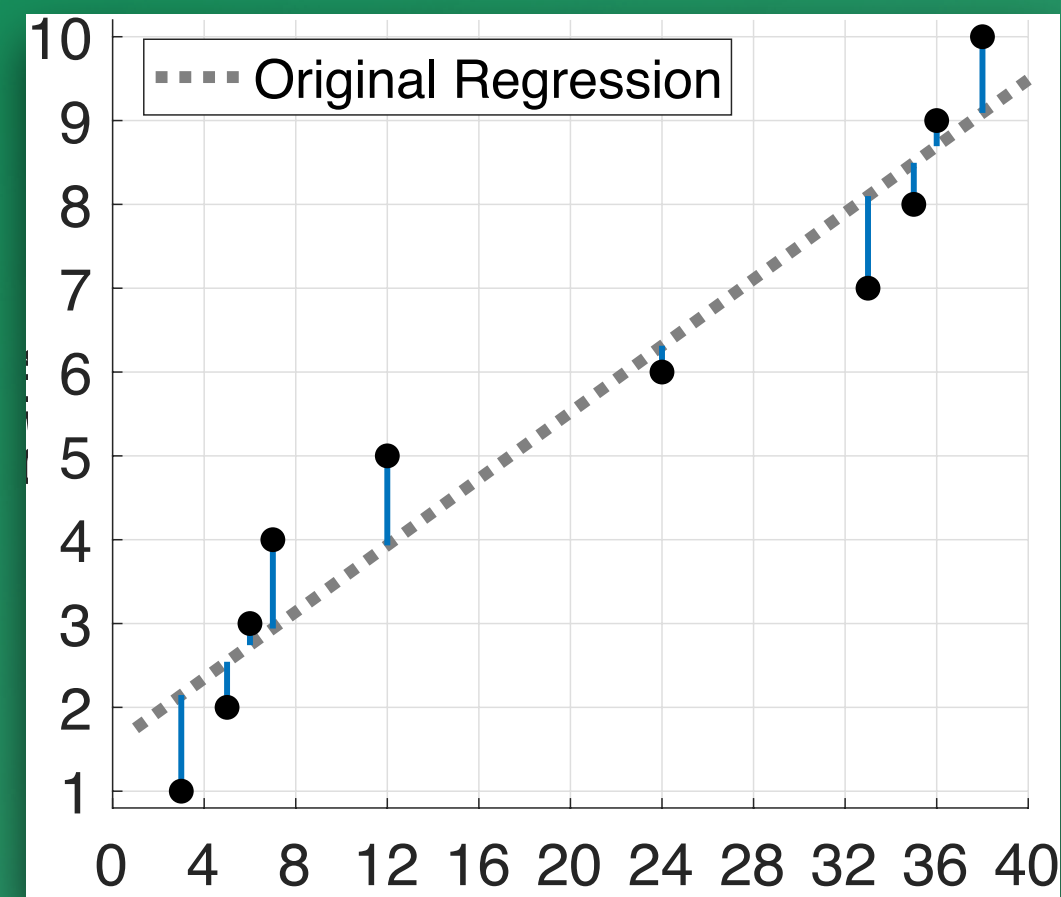
Attacks on  
CDF Regression

# Part-3



Attacks on  
Hierarchical Learned Index

# Part-1



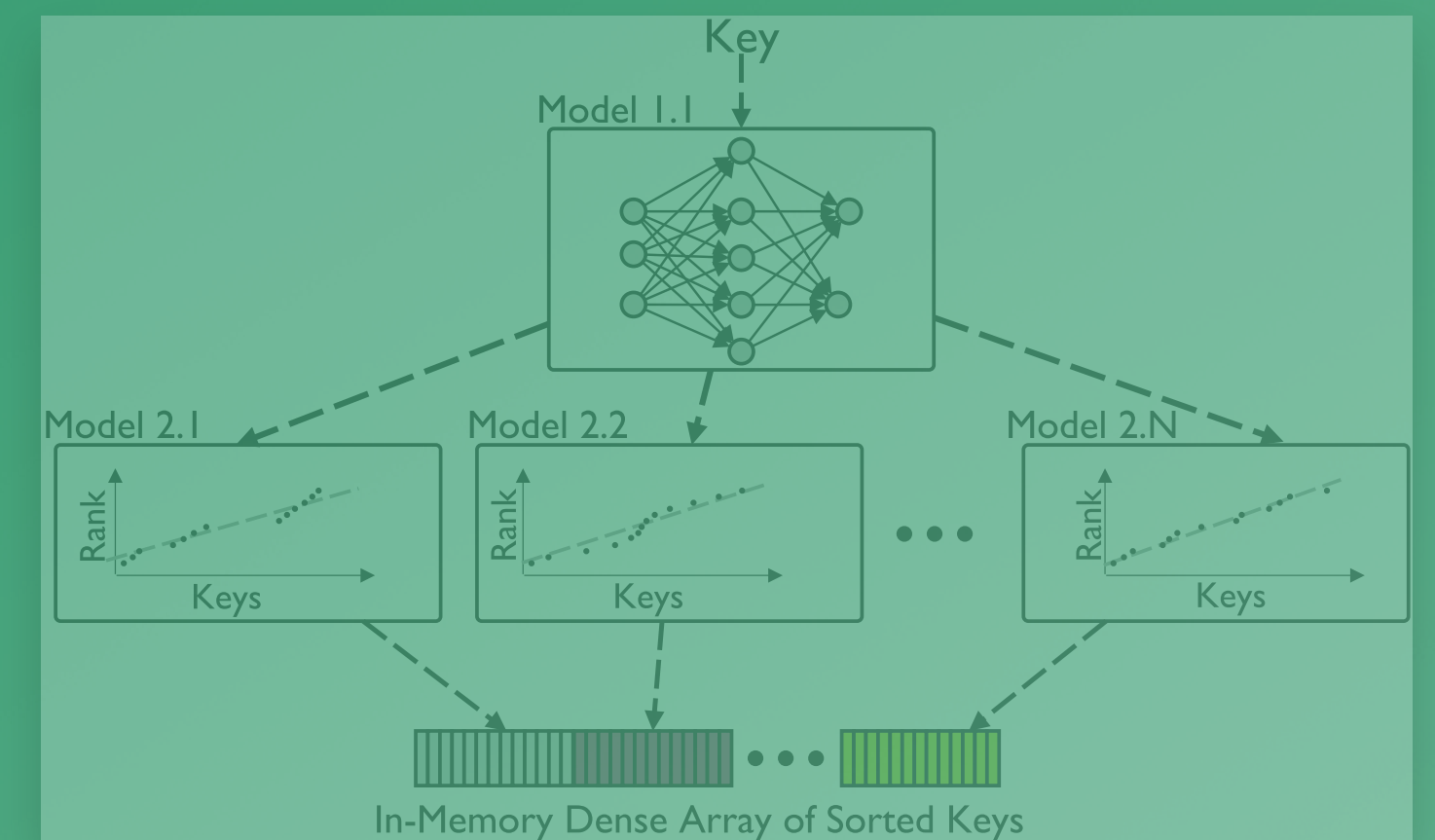
Attacks on  
(Vanilla) Regression

# Part-2



Attacks on  
CDF Regression

# Part-3

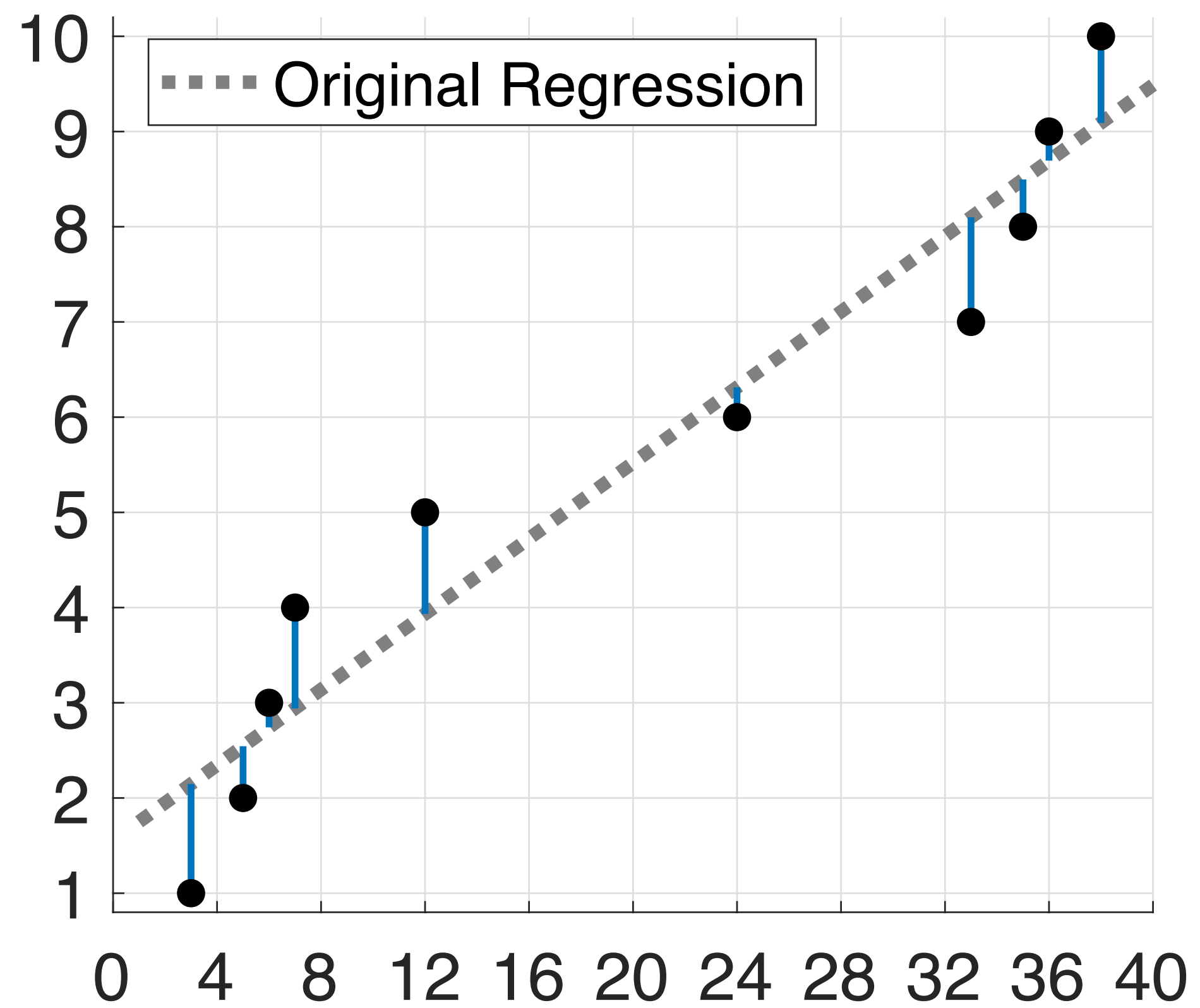


Attacks on  
Hierarchical Learned Index



# WHAT IS REGRESSION

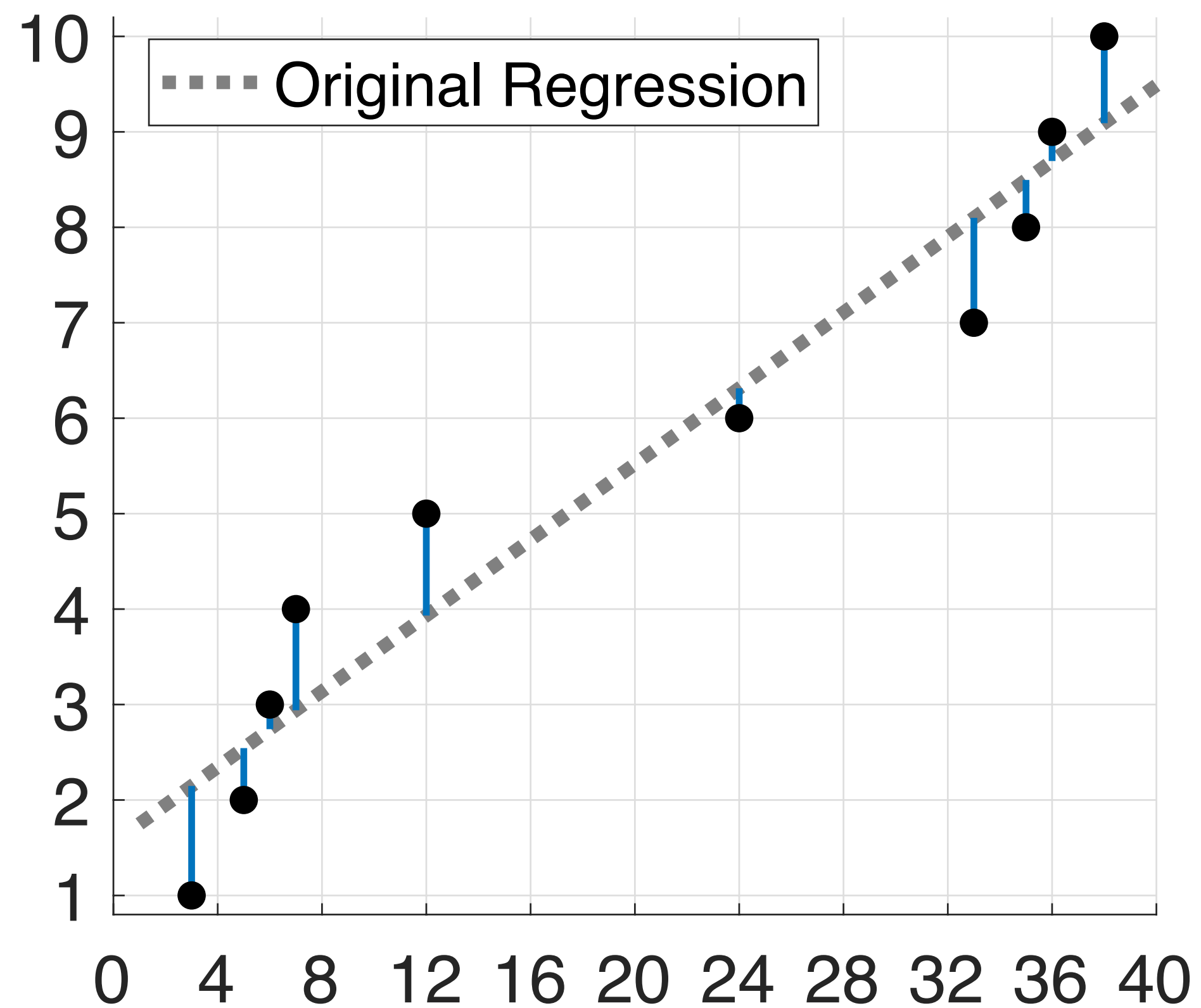
## (VANILLA) MODEL





# WHAT IS REGRESSION

## KNOWN POISONING APPROACHES



### THREAT MODEL

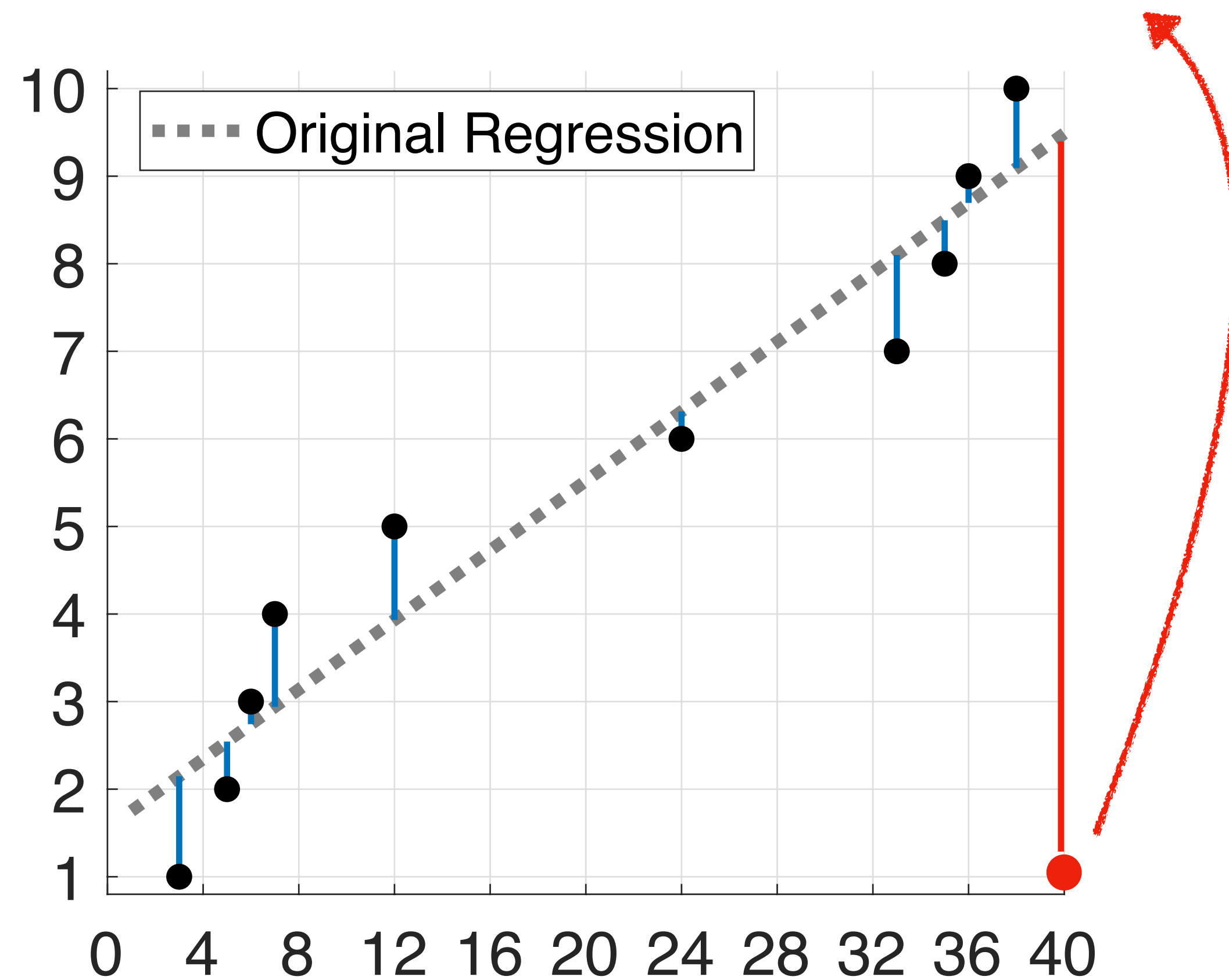
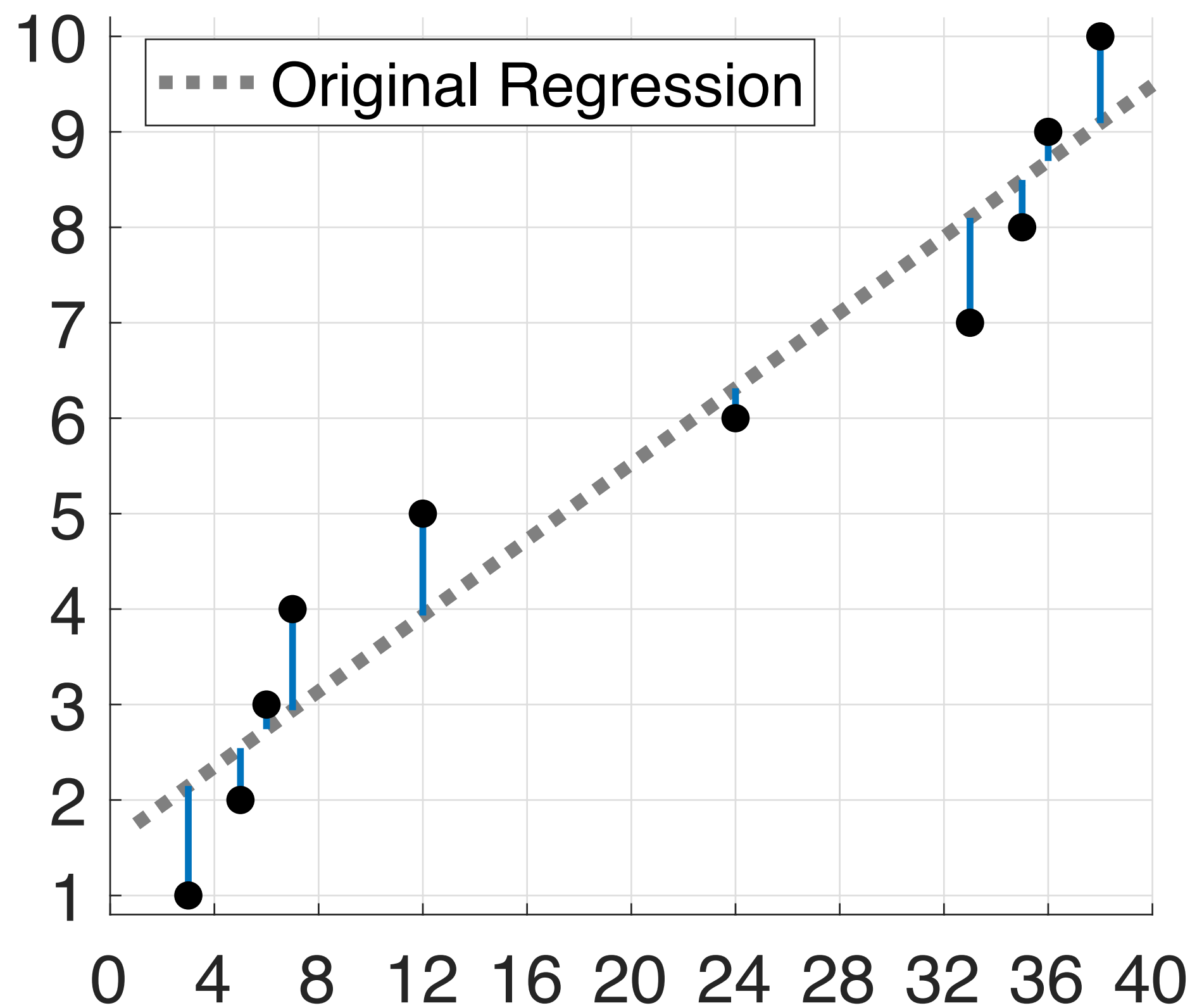
- Attacker: Malicious **contribution** points
- Power: Access to data (**White-box** Attack)
- Goal: **Degrade** Performance



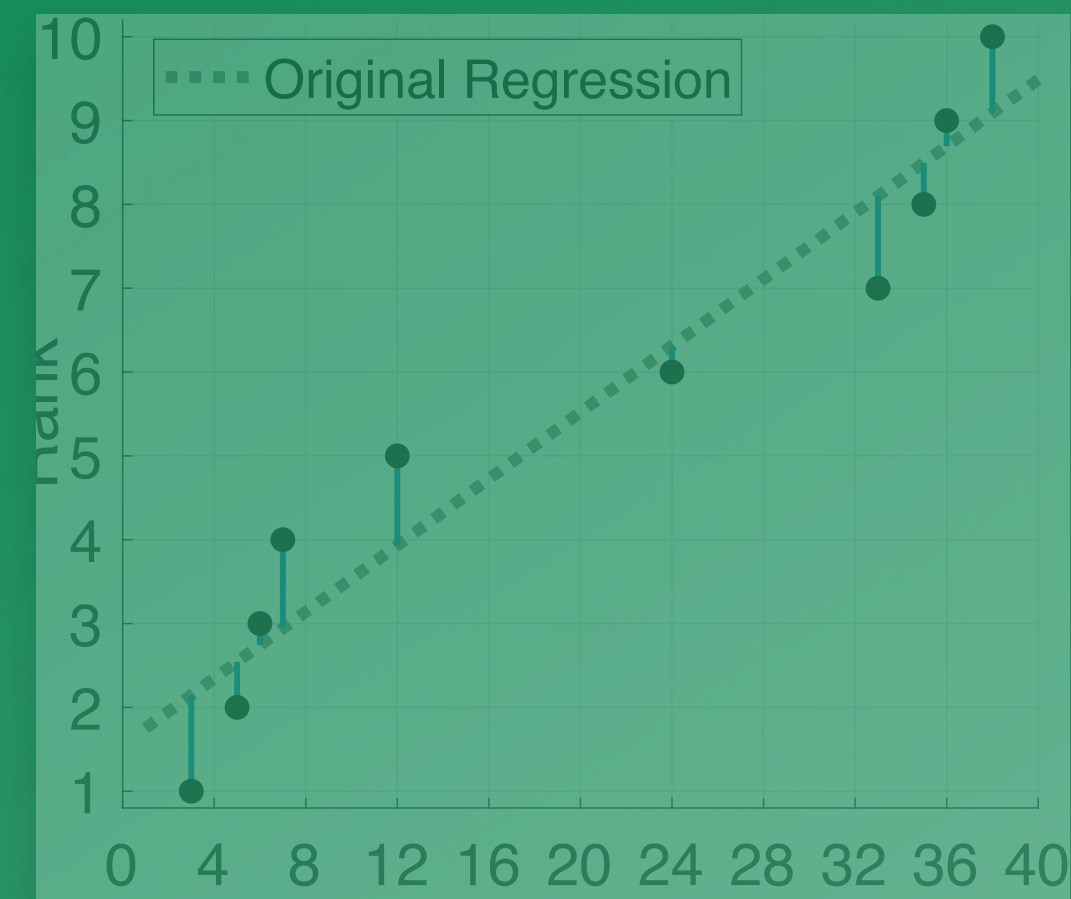
# (VANILLA) REGRESSION MODEL

## KNOWN POISONING APPROACHES

Poisoning point can be **any** (X,Y)

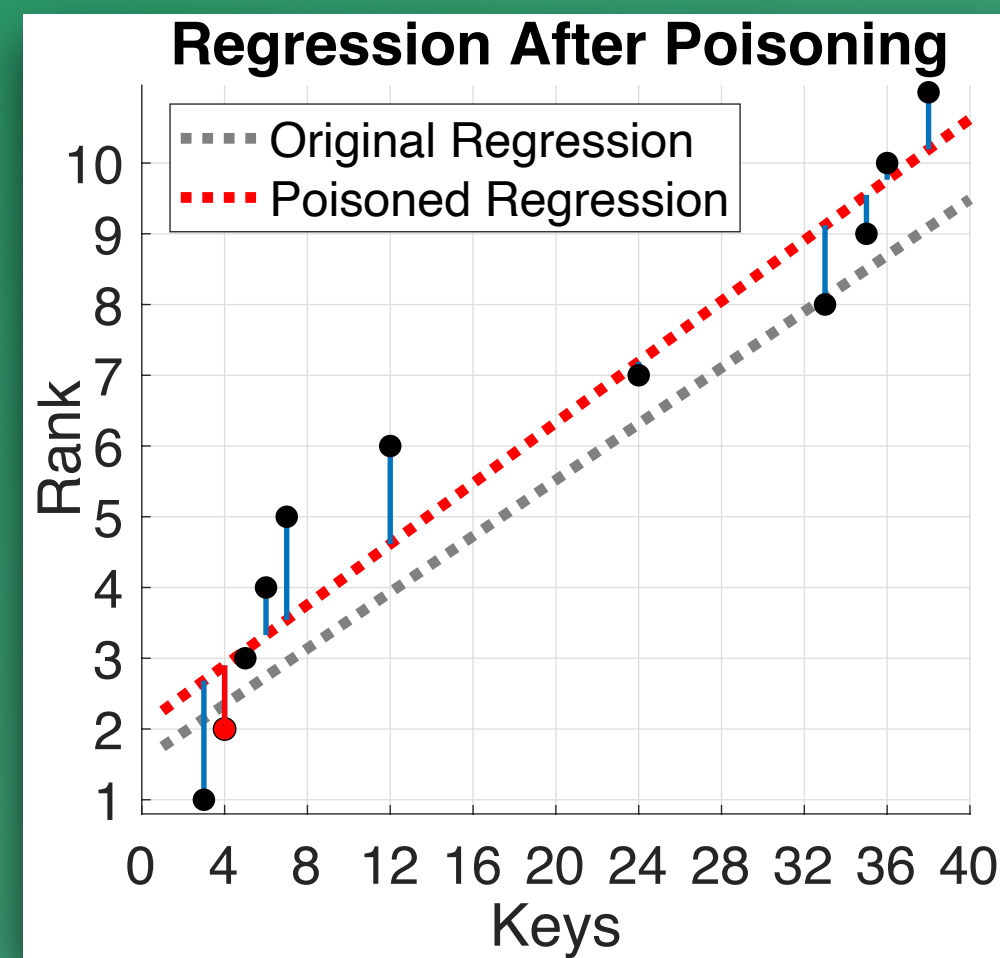


# Part-1



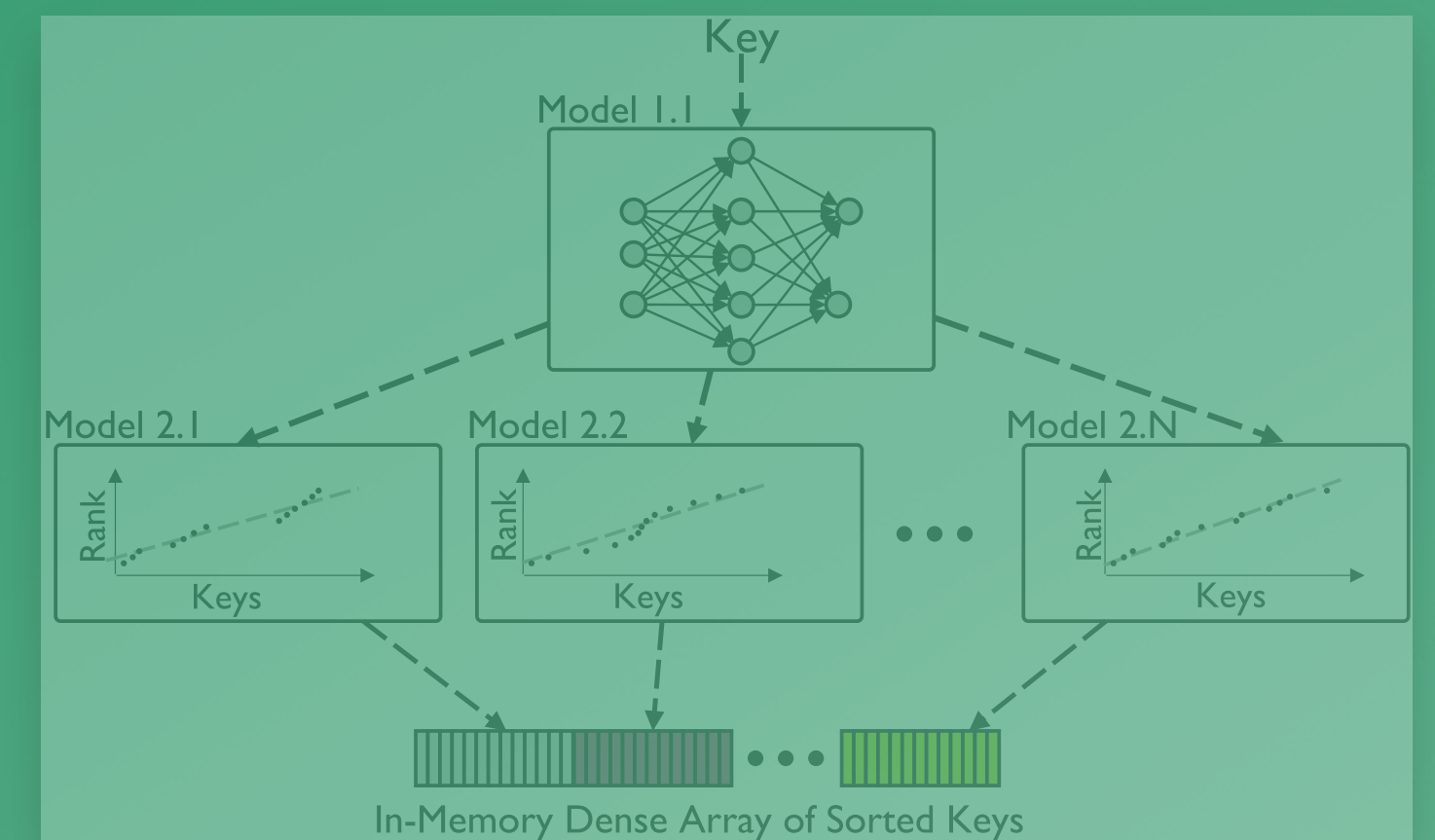
Attacks on  
(Vanilla) Regression

# Part-2



Attacks on  
CDF Regression

# Part-3

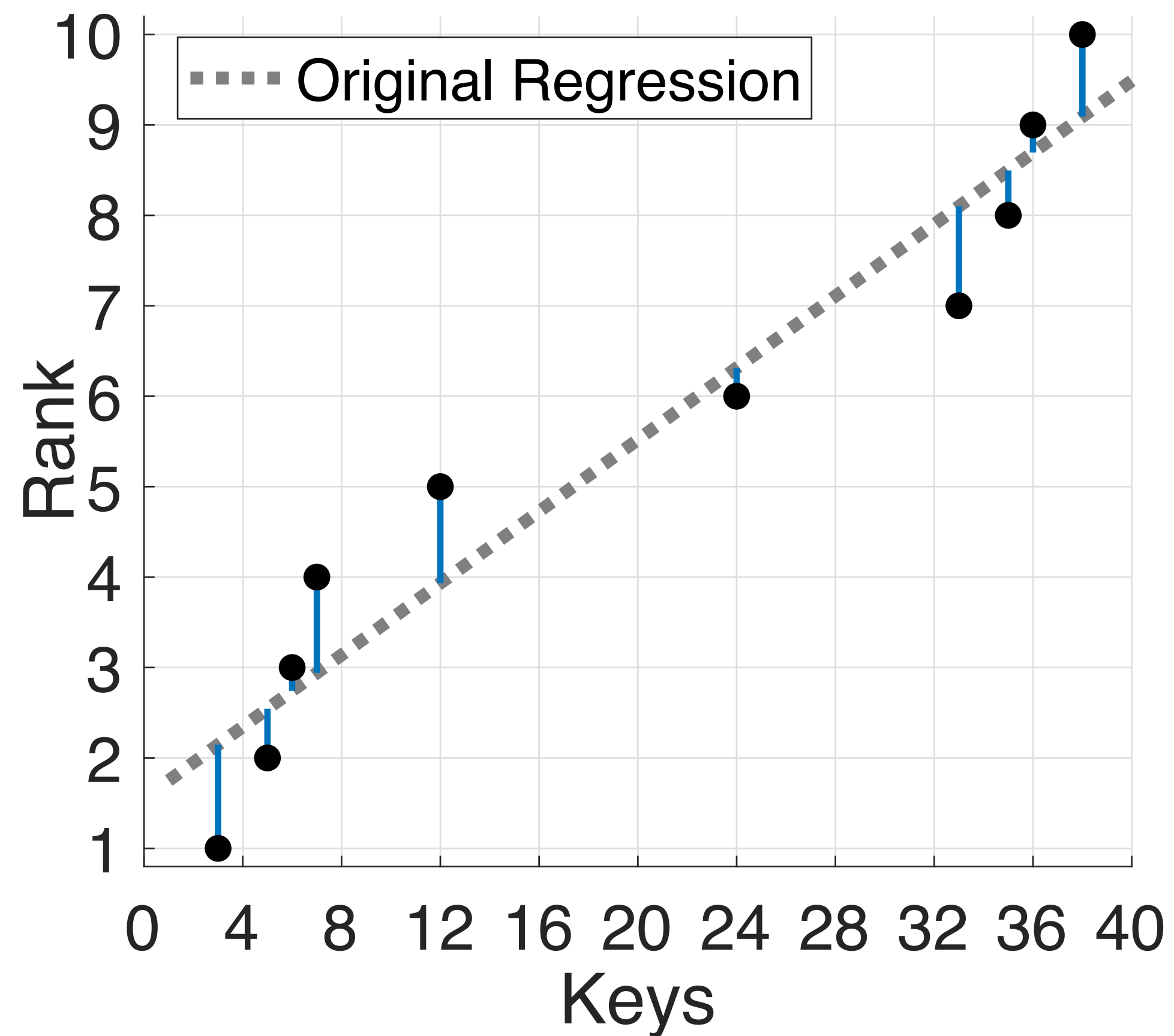


Attacks on  
Hierarchical Learned Index



# REGRESSION IN THE CONTEXT OF LIS

## WHAT'S THE DIFFERENCE?

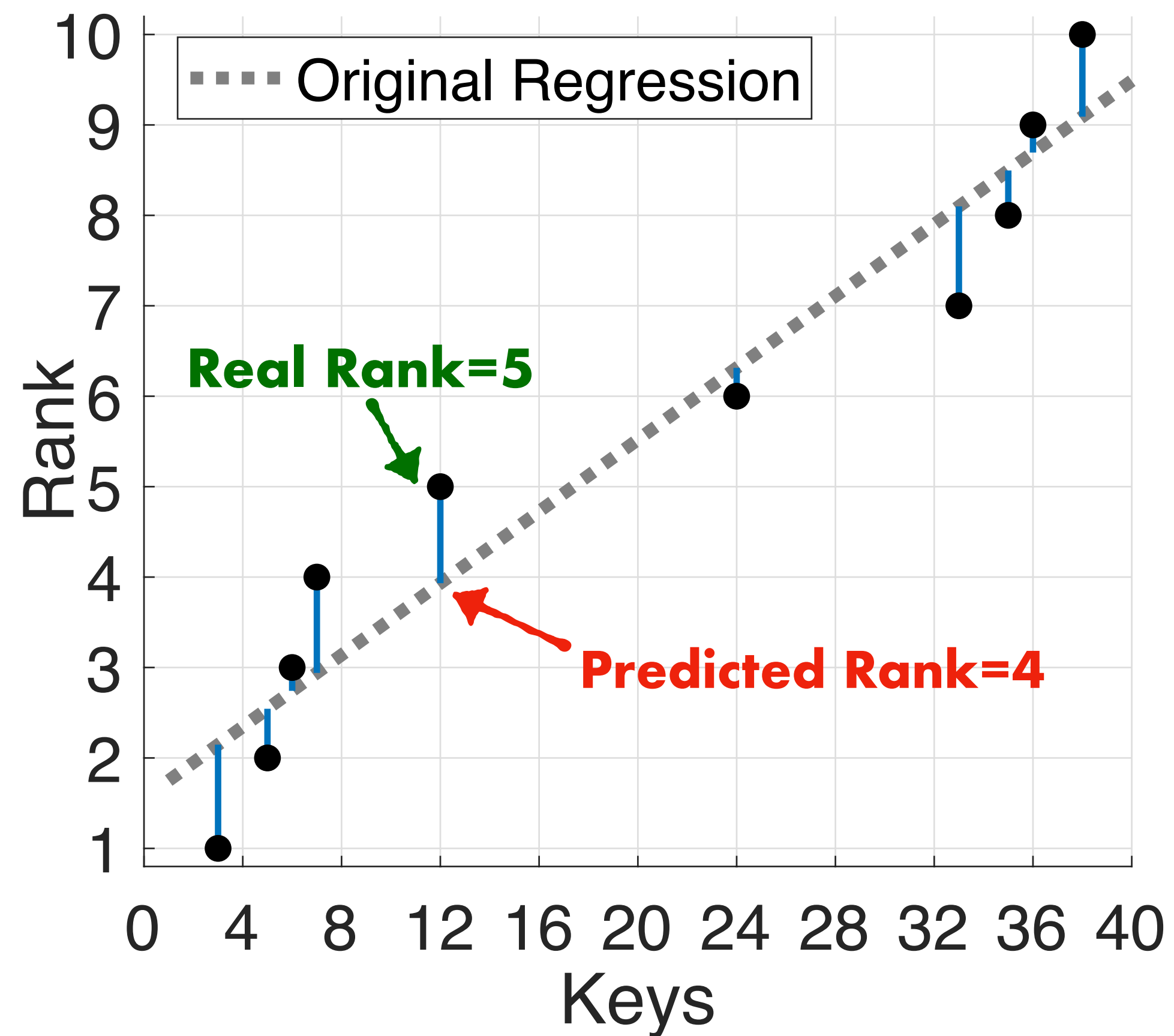


**Cumulative Density Function**



# REGRESSION MODEL ON CDFs

## IMPACT OF AN ERROR



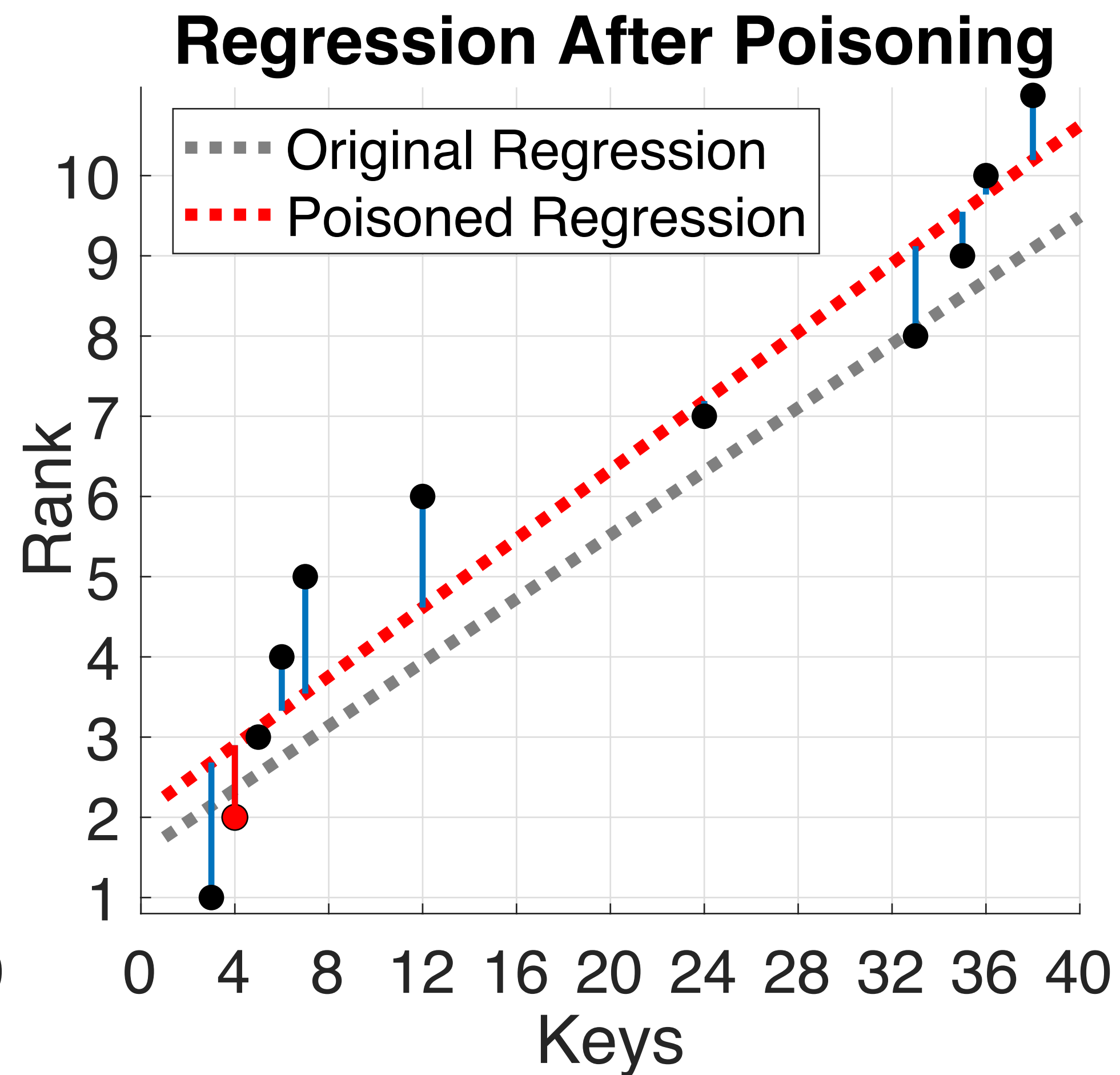
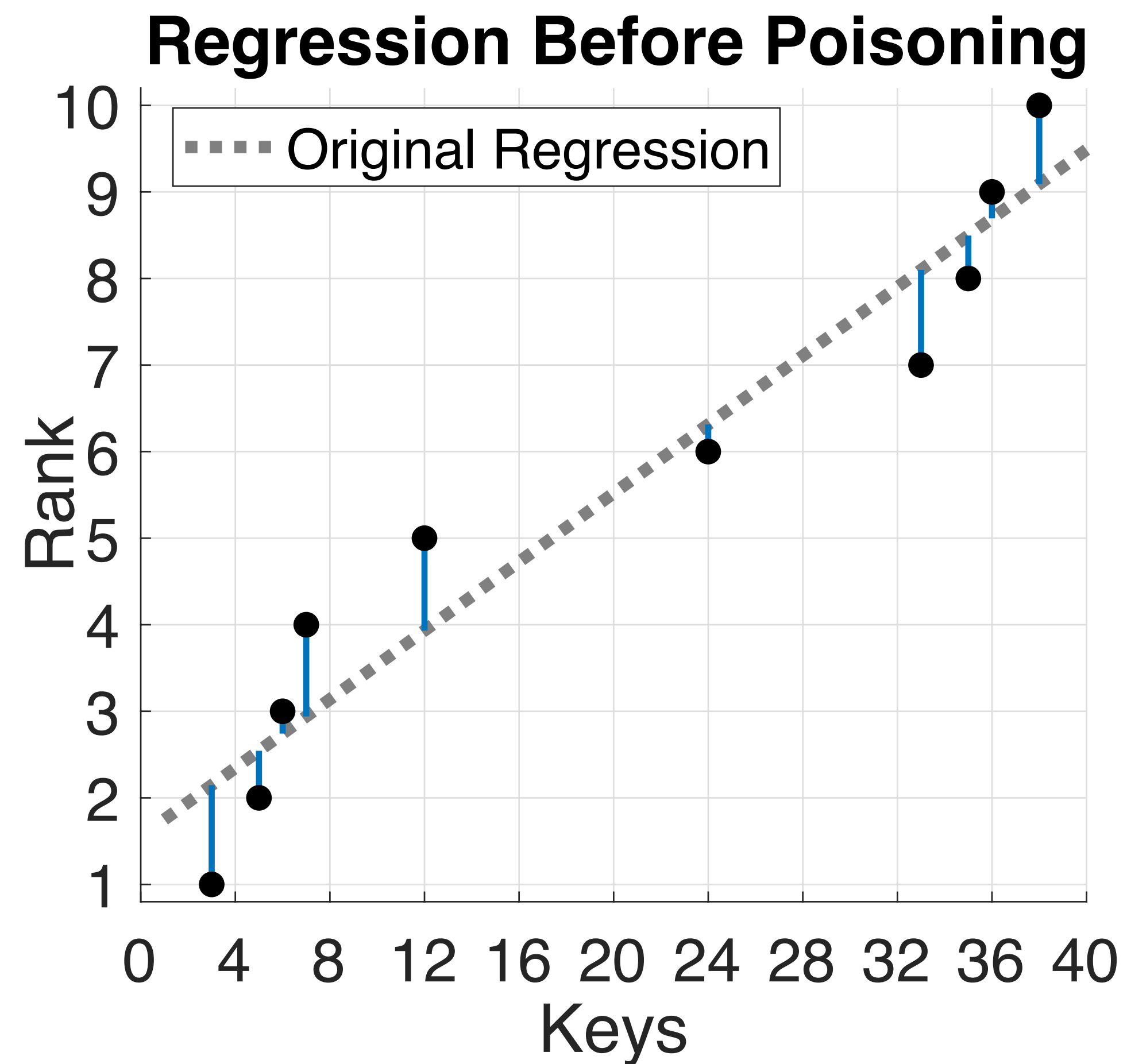
**Prediction Error:  
Trigger Local Search to Fix**

Cumulative Density Function



# REGRESSION MODEL ON CDFs

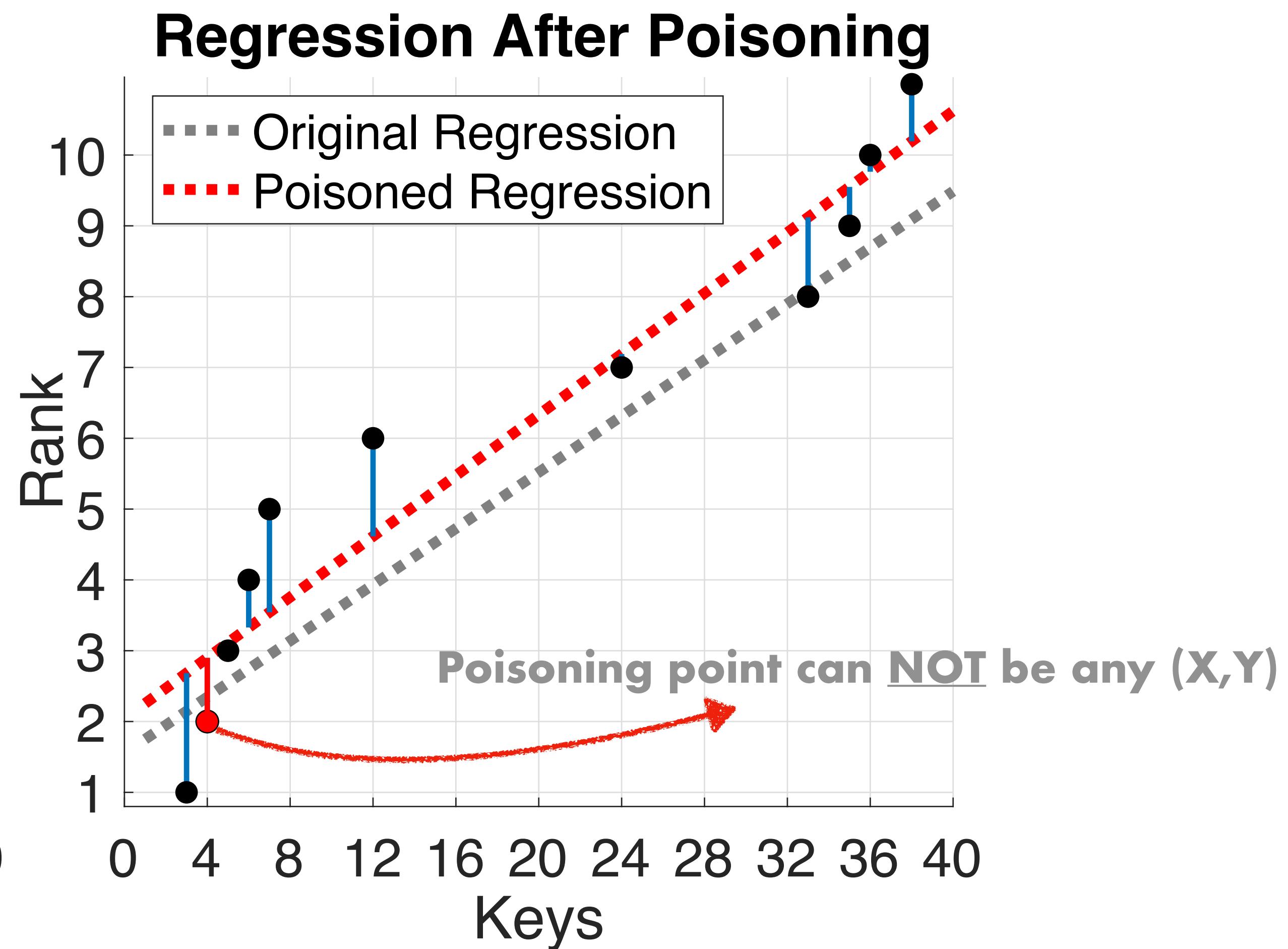
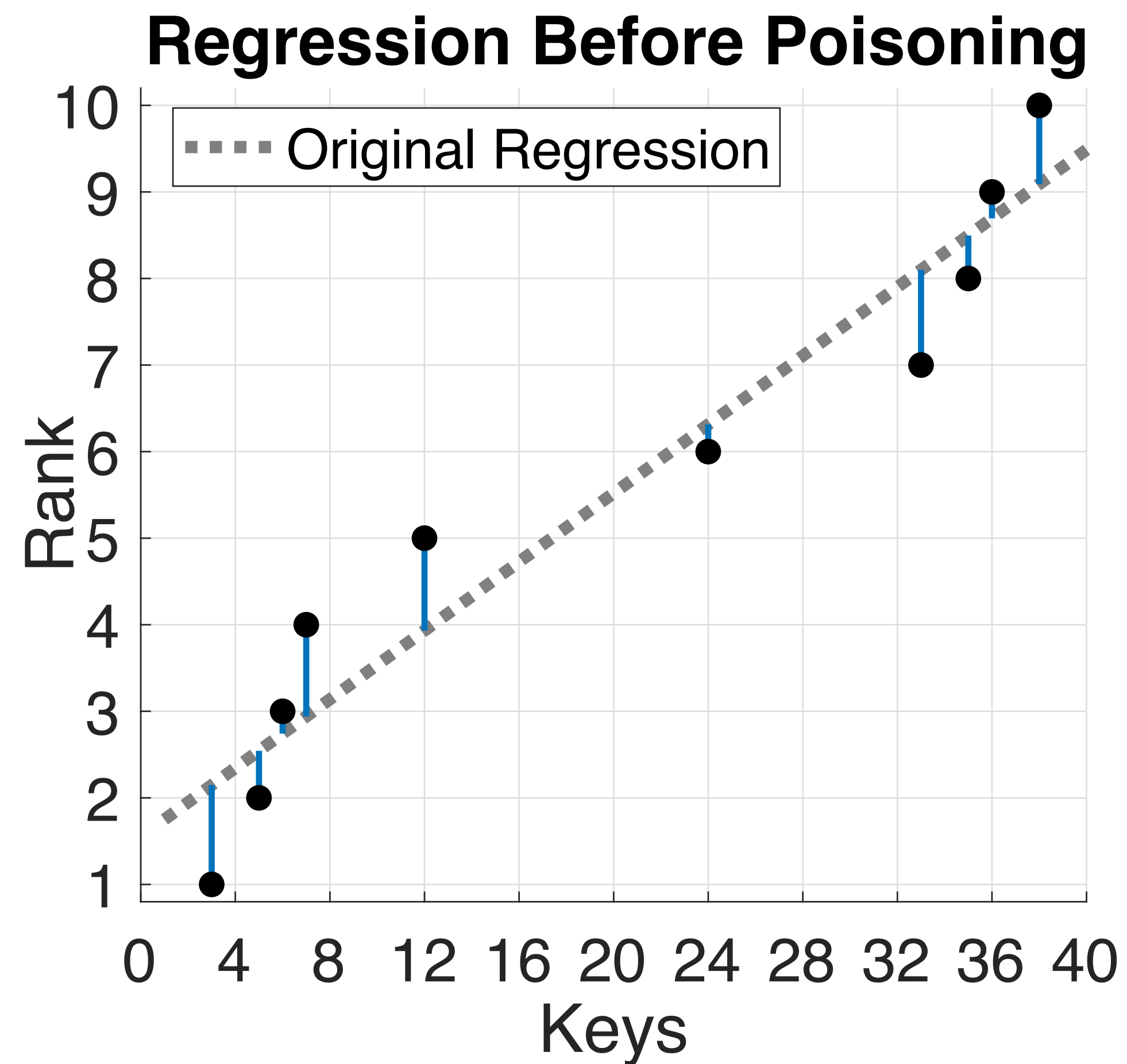
## POISONING CDFs





# REGRESSION MODEL ON CDFs

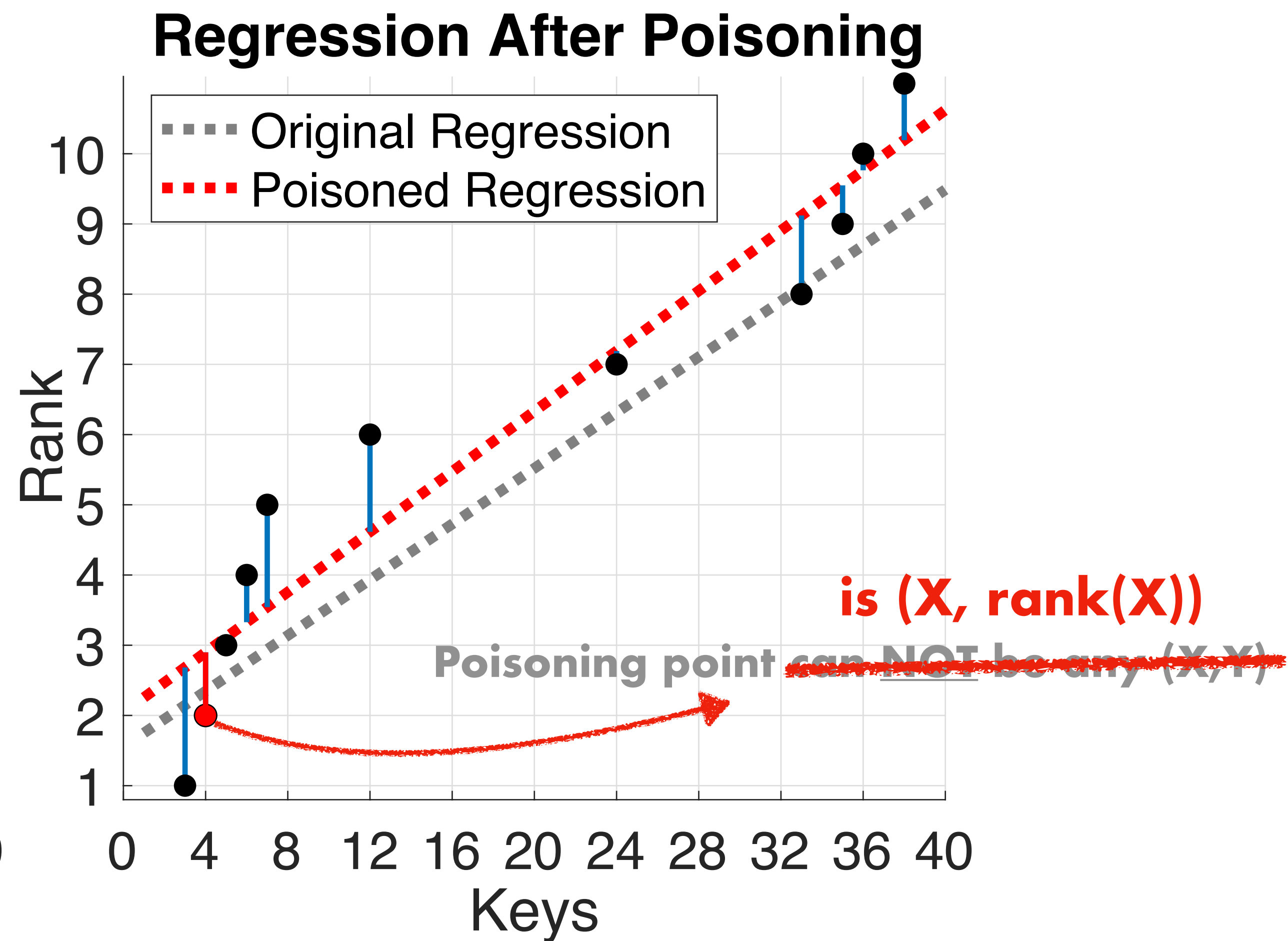
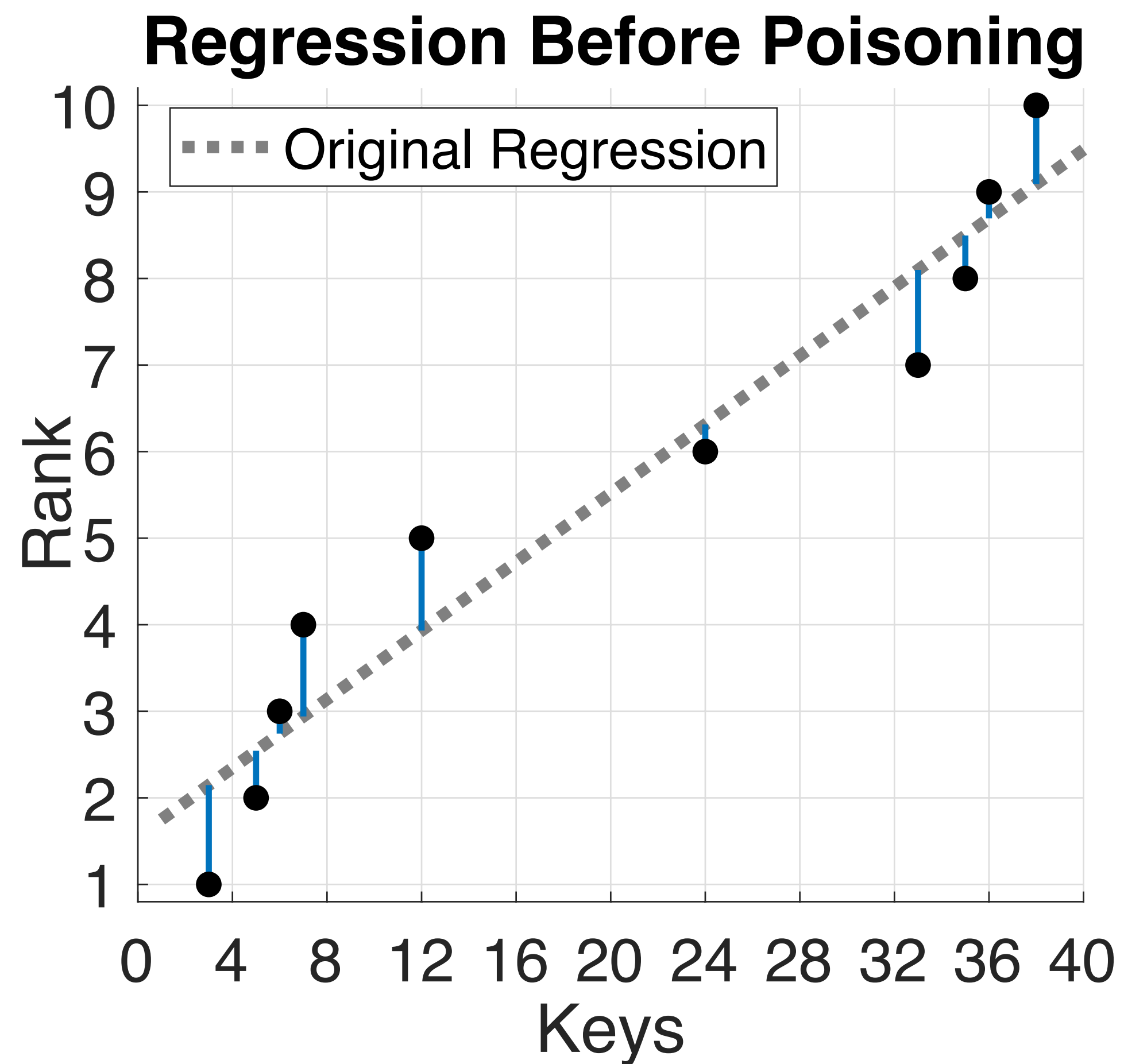
## POISONING CDFs





# REGRESSION MODEL ON CDFs

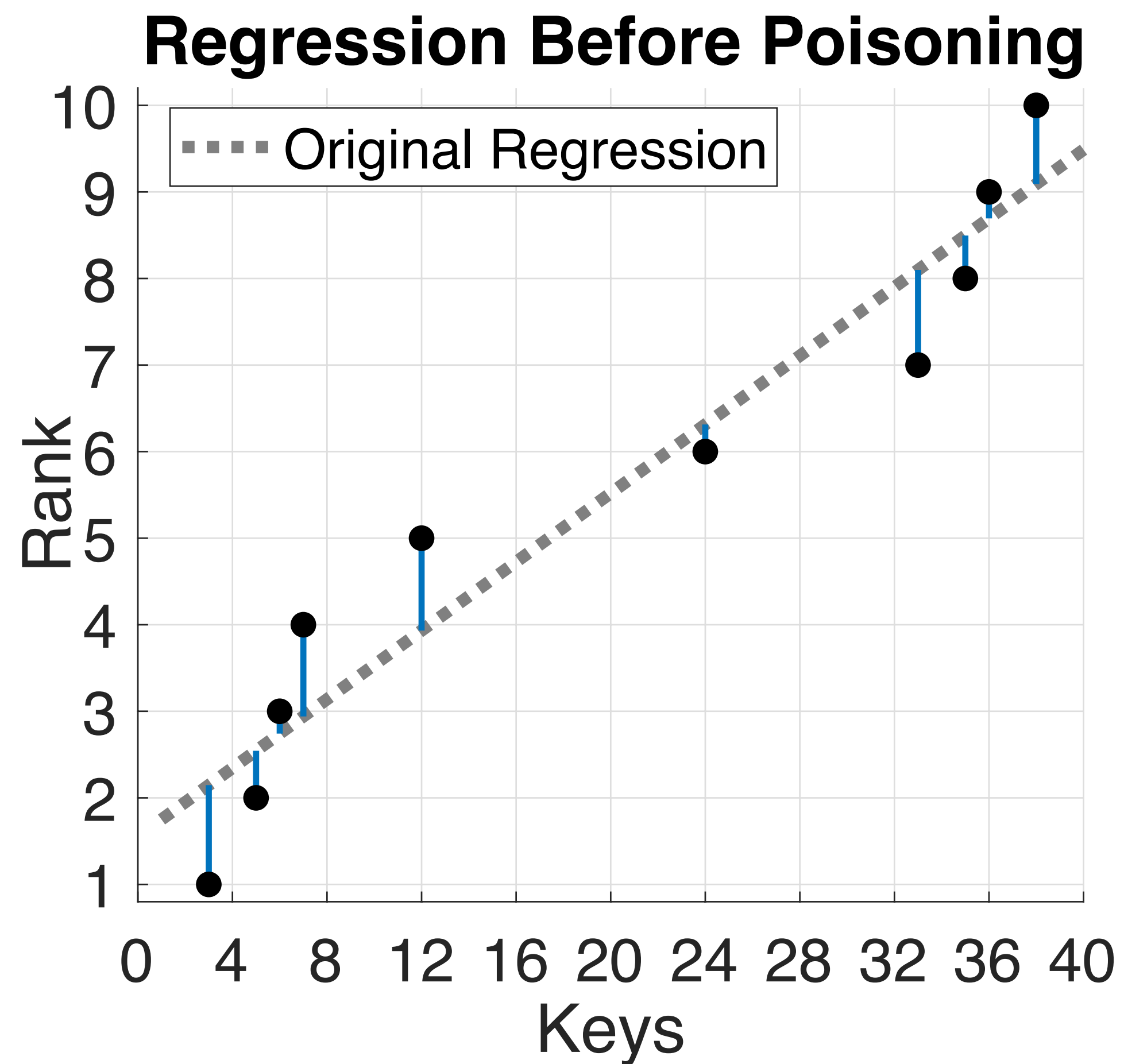
## WHAT'S THE DIFFERENCE?





# REGRESSION MODEL ON CDFs

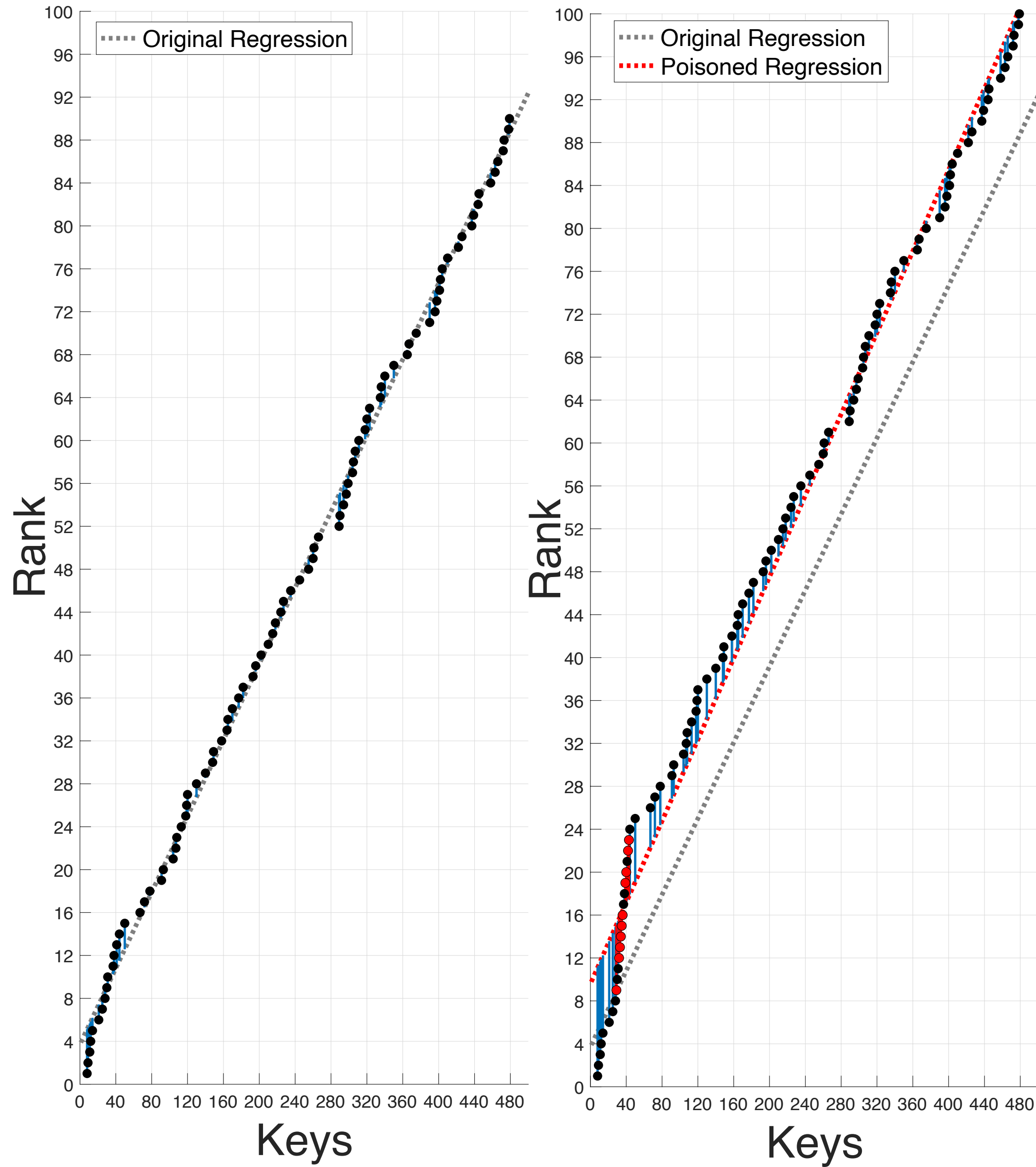
## WHAT'S THE DIFFERENCE?





# REGRESSION MODEL ON CDFs

## OVERVIEW



## CONTRIBUTIONS

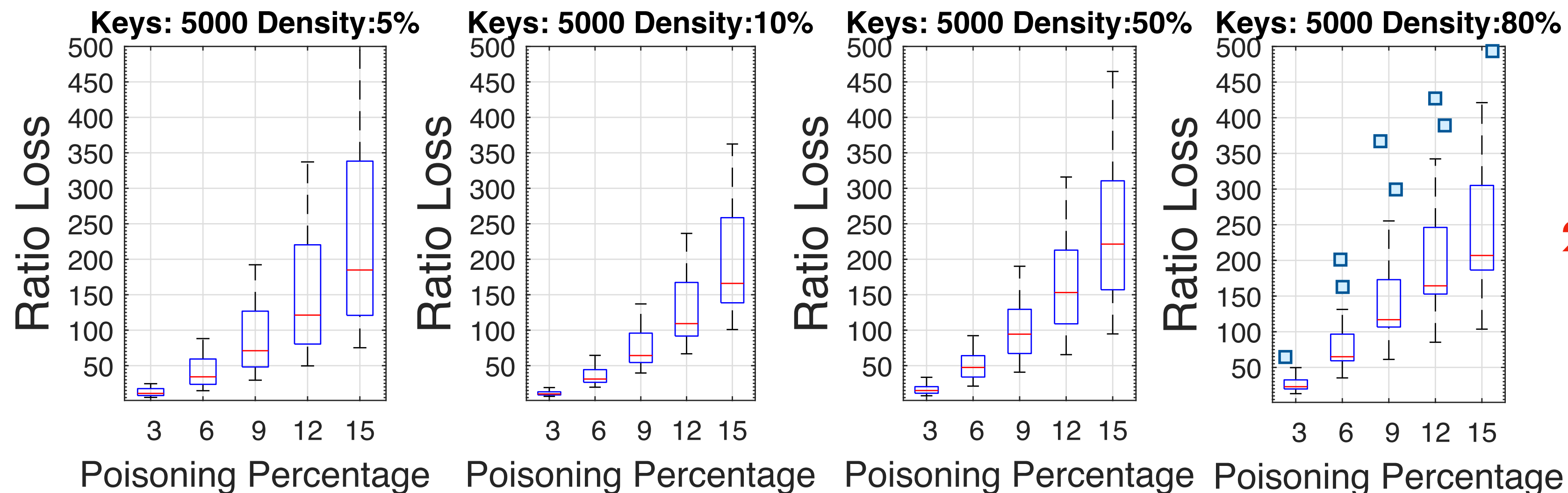
- Give a linear poisoning attack for a single point
- Greedy poisoning attack for multiple points
- Poisoning percentage  $< 15\%$
- Evaluation Metrics:
  - $\text{Ratio Loss} = \text{Poisoned\_MSE} / \text{MSE}$
  - $\text{Memory Offset} = \text{Predict. Location} - \text{Real Location}$



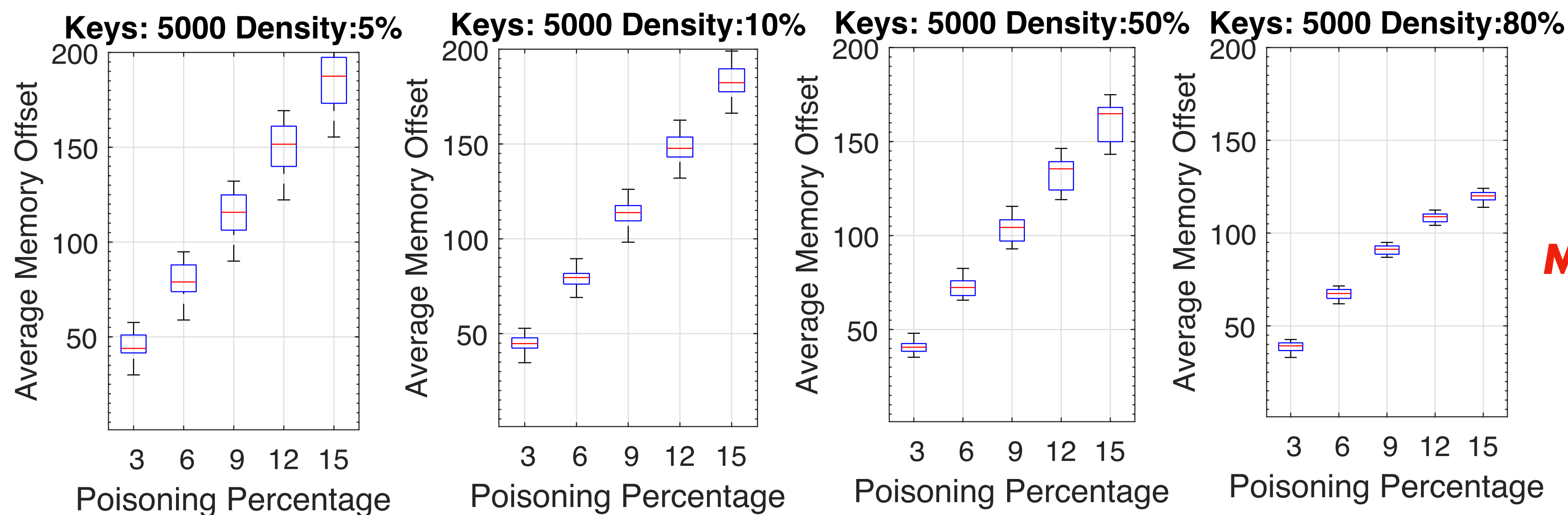
# REGRESSION MODEL ON CDFs

## EVALUATION

### Uniform Key Distribution



**200x larger MSE**



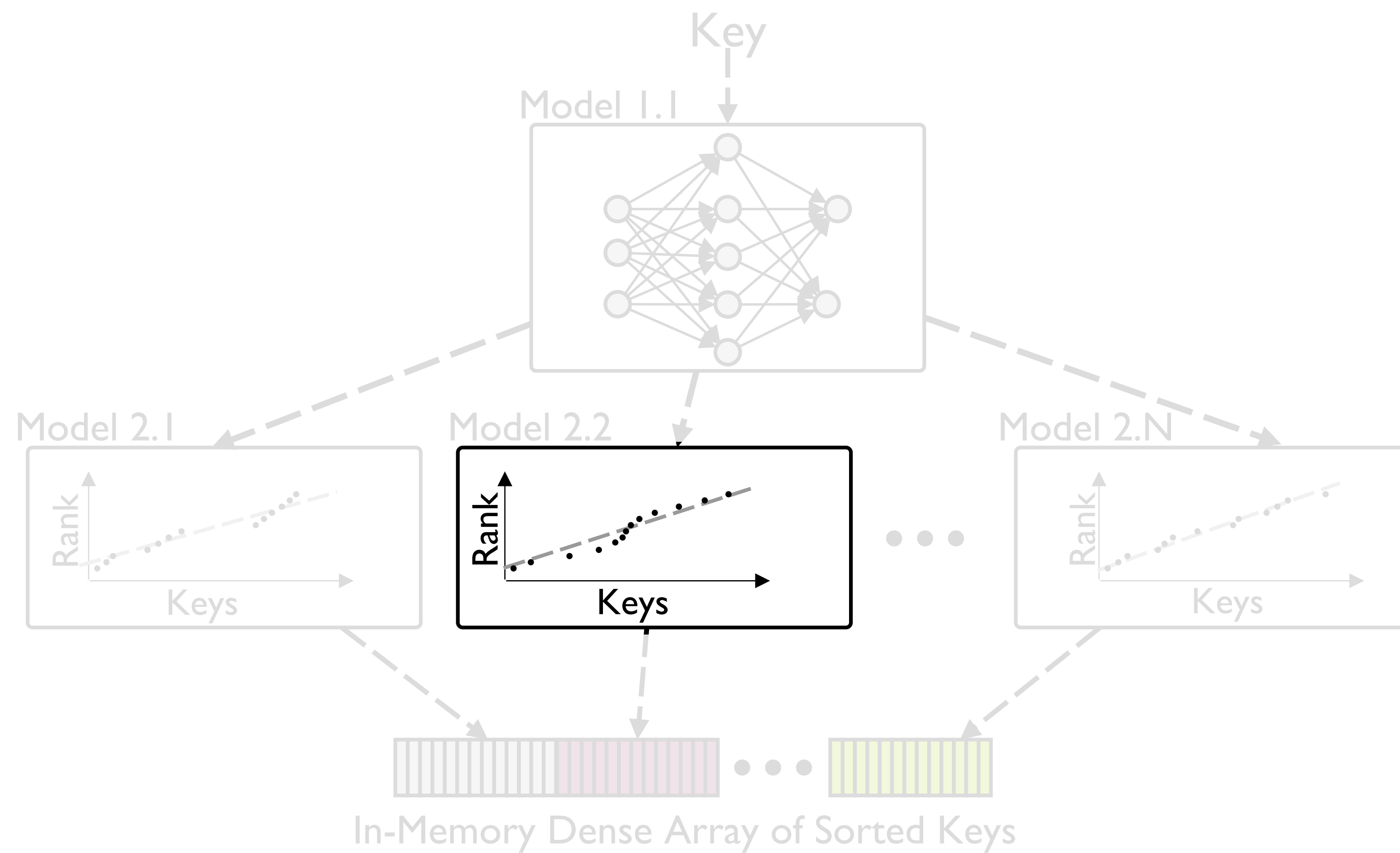
**Memory Offset 125-180**





# ATTACK SO FAR...

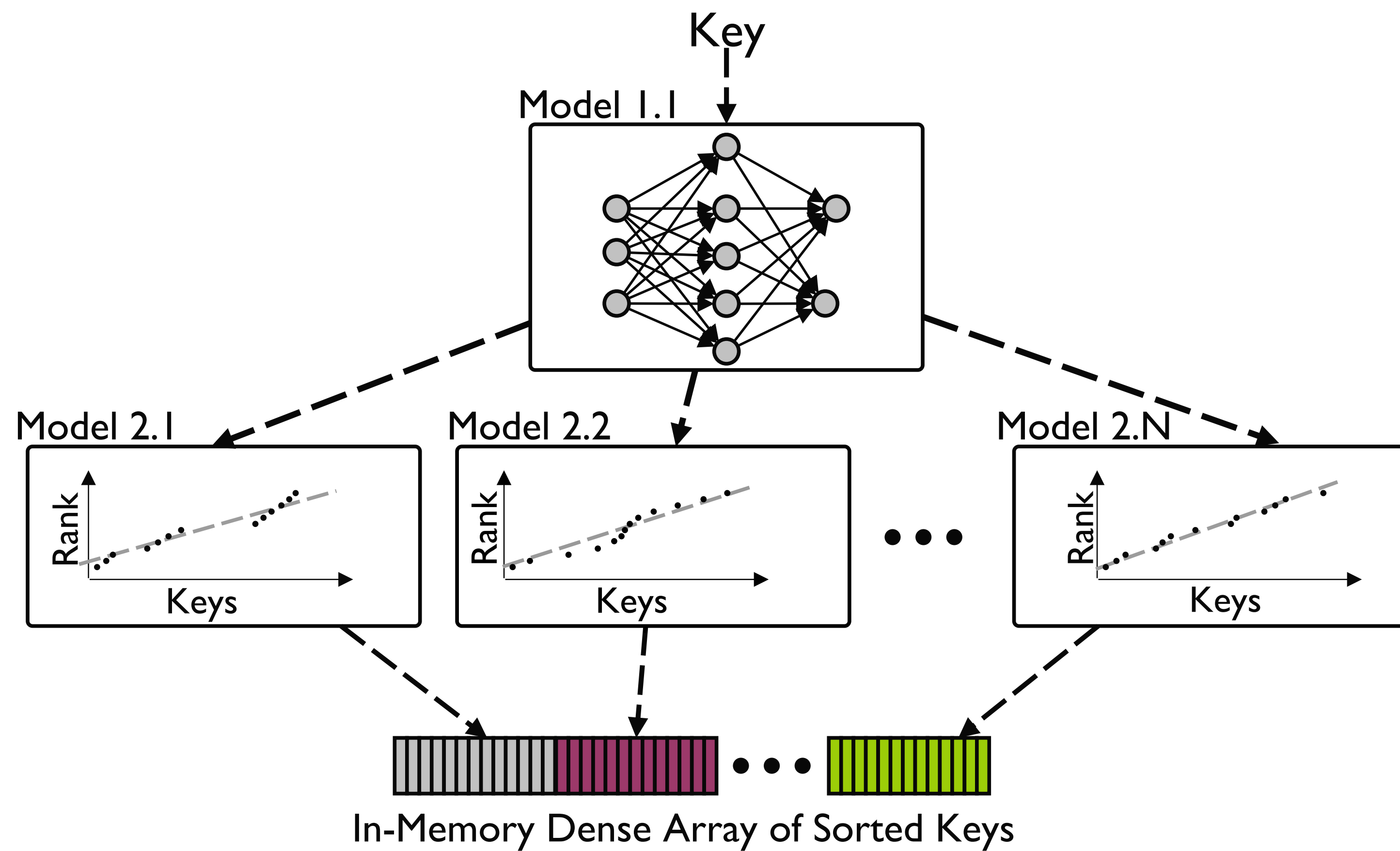
## ONLY REGRESSION ON CDF





# HIERARCHICAL MODELS

## TWO-STAGE ARCHITECTURE



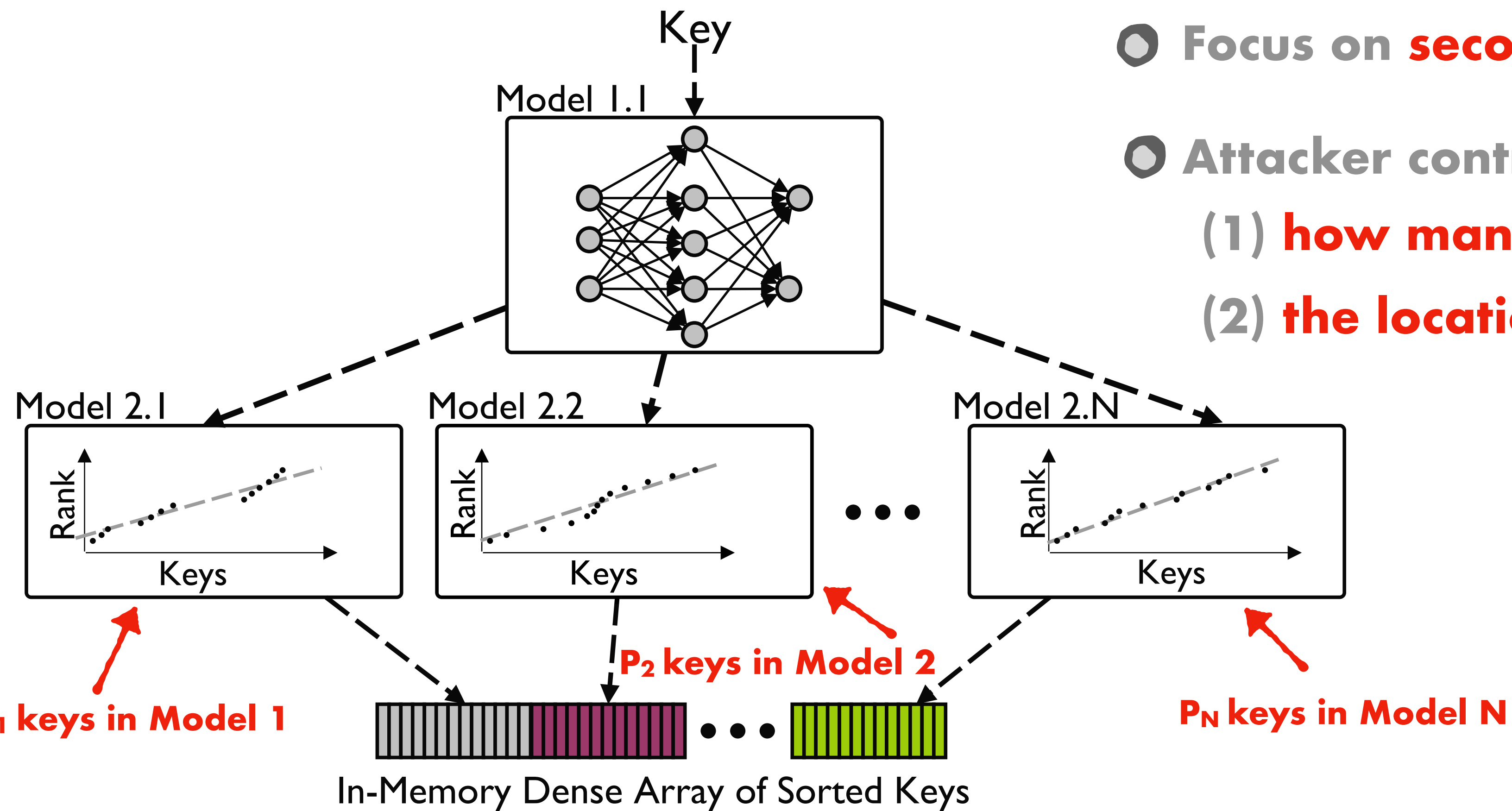


# ATTACK ON HIERARCHICAL MODELS

## TWO-STAGE ARCHITECTURE

### ADVERSARIAL APPROACH

- Focus on **second-level** poisoning (regression)
- Attacker controls
  - (1) **how many** keys per model (Volume)
  - (2) **the location** of poisoning keys per model



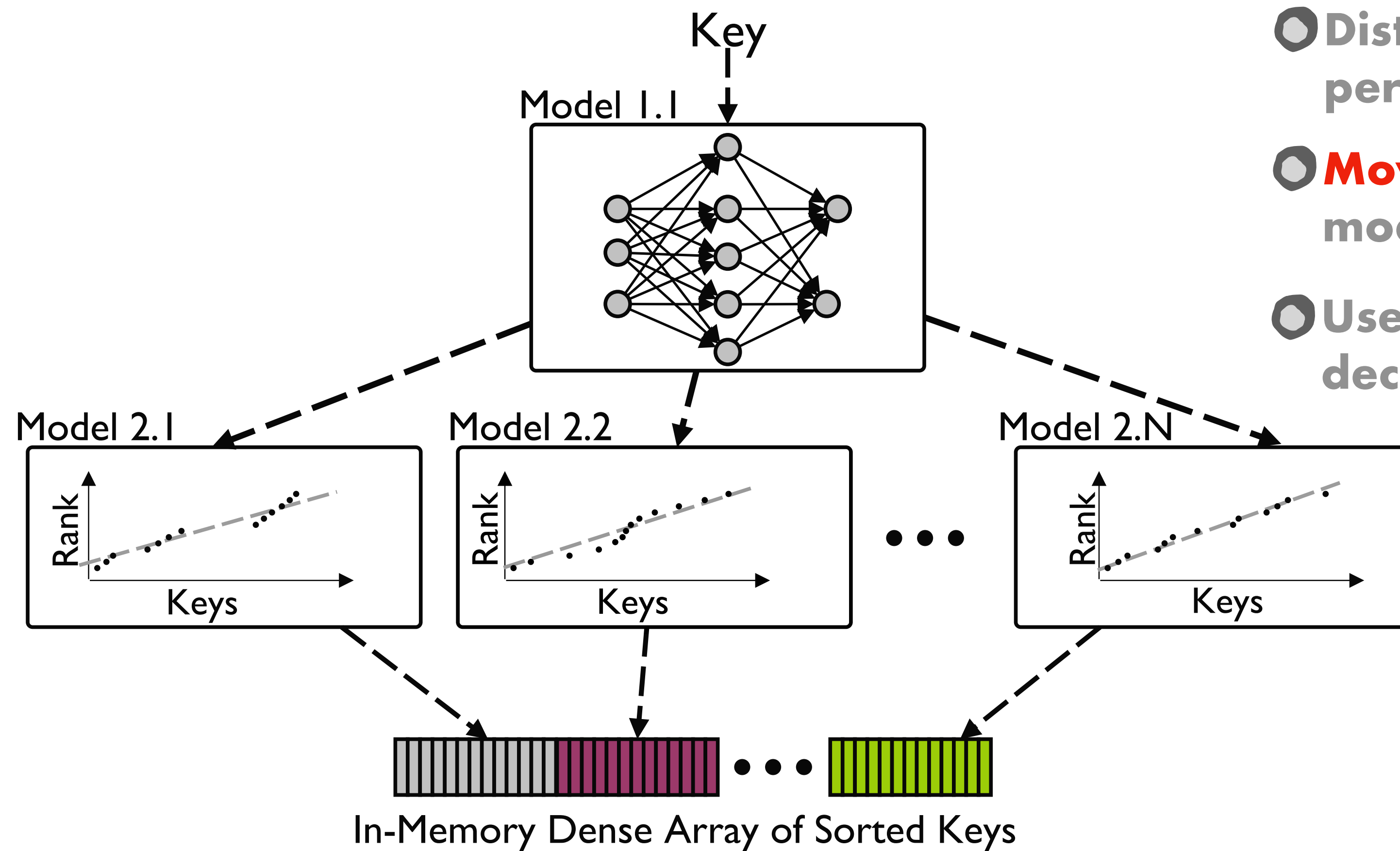


# ATTACK ON HIERARCHICAL MODELS

## TWO-STAGE ARCHITECTURE

### (HIGH-LEVEL) ALGORITHM

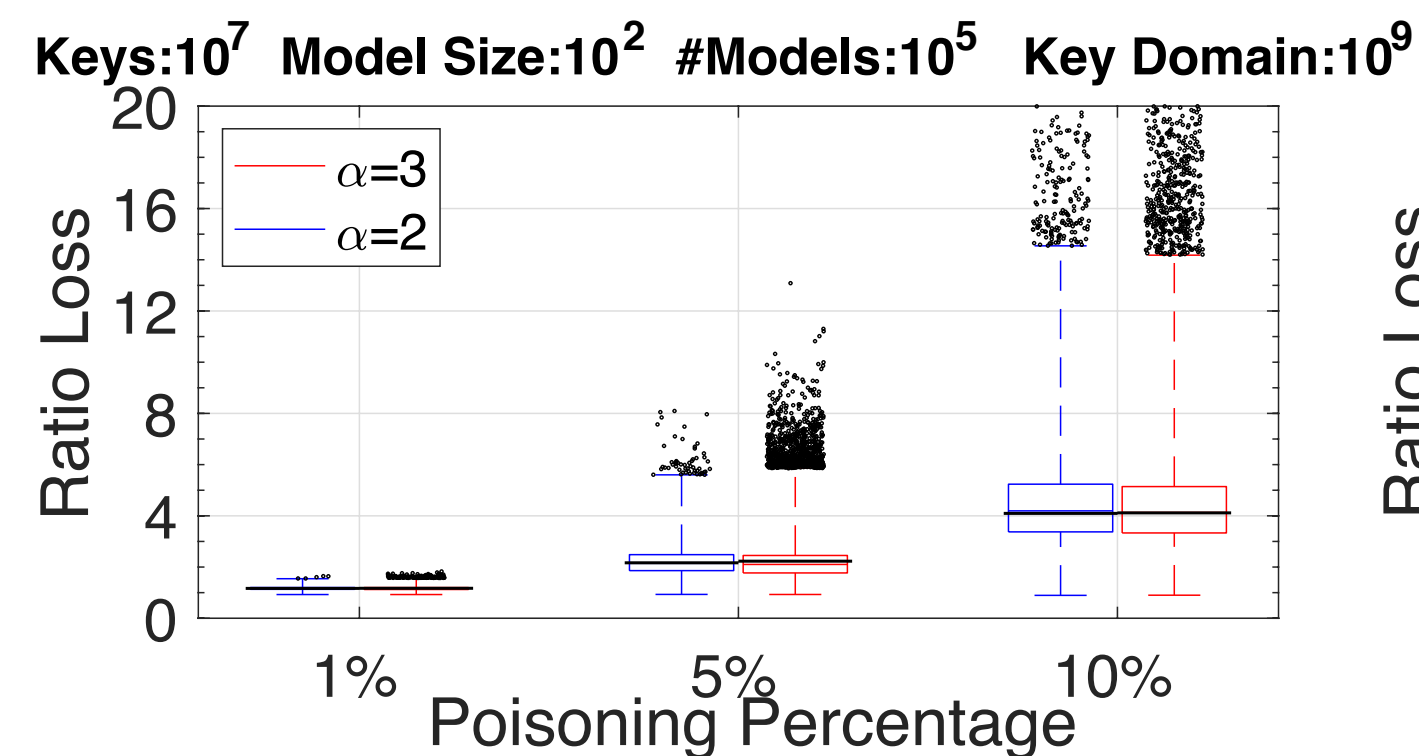
- Distribute the **same number** of poisoning keys per model
- **Move** a poisoning key to the next/previous model if it increases the total error
- Use **previous multipoint regression attack** to decide which poisoning points to insert



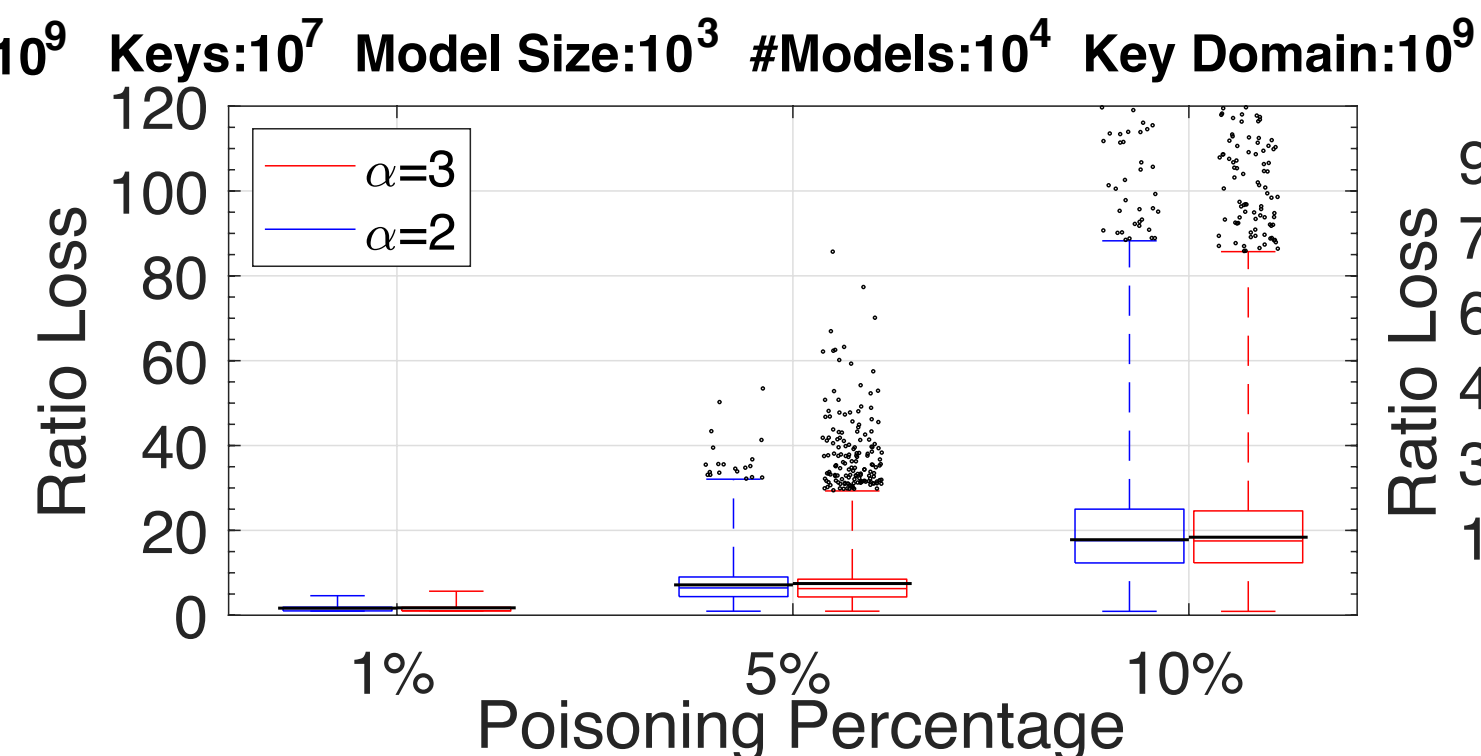


# ATTACK ON HIERARCHICAL MODELS EVALUATION

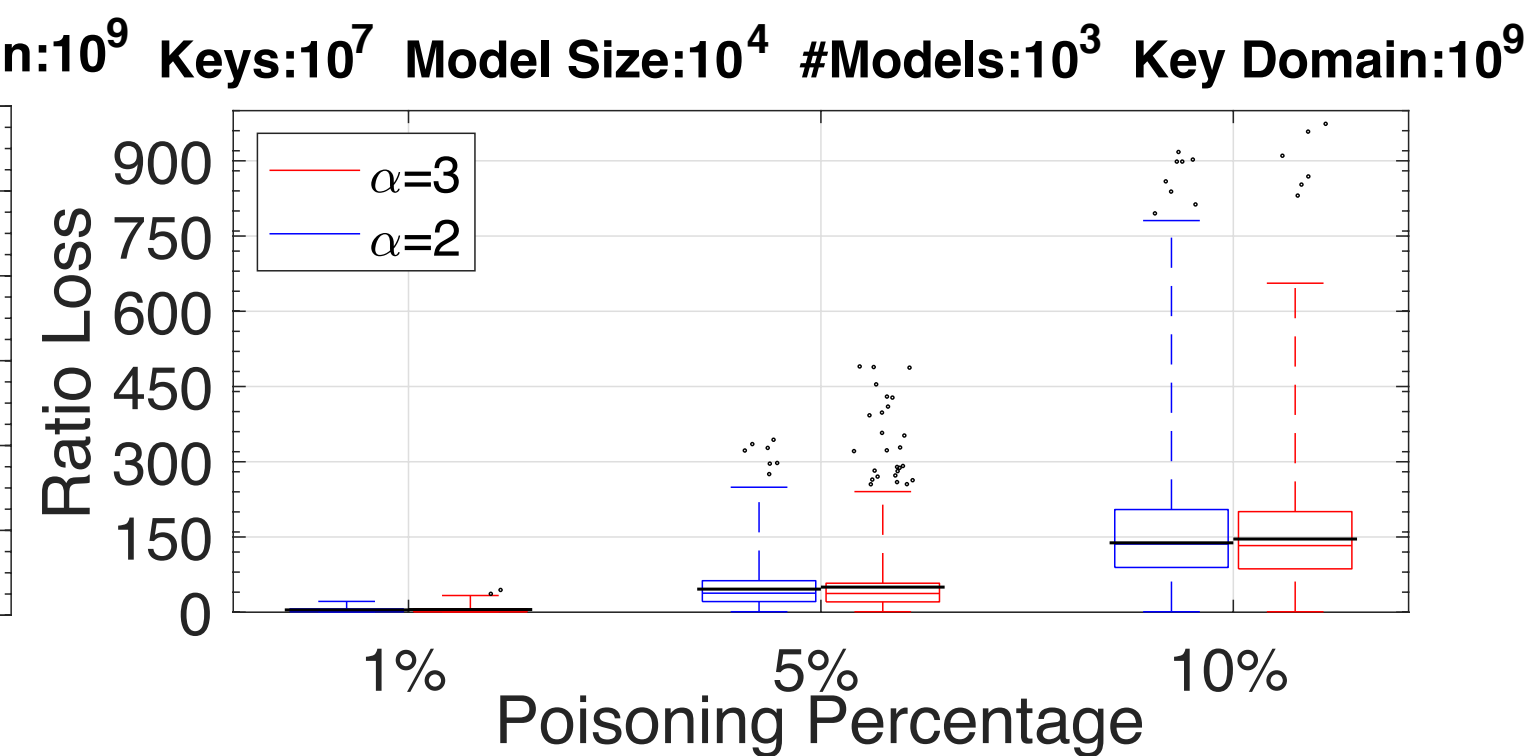
## Uniform Key Distribution



**4x larger MSE**

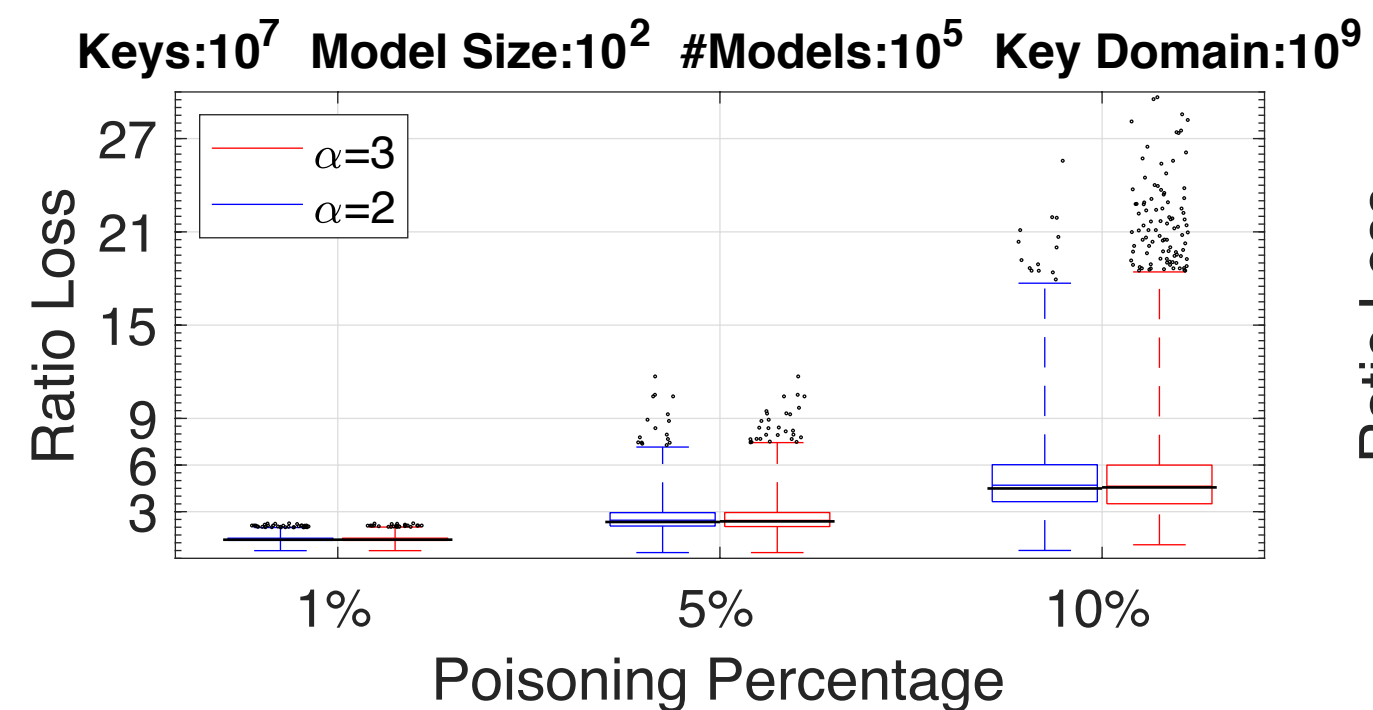


**20x larger MSE**

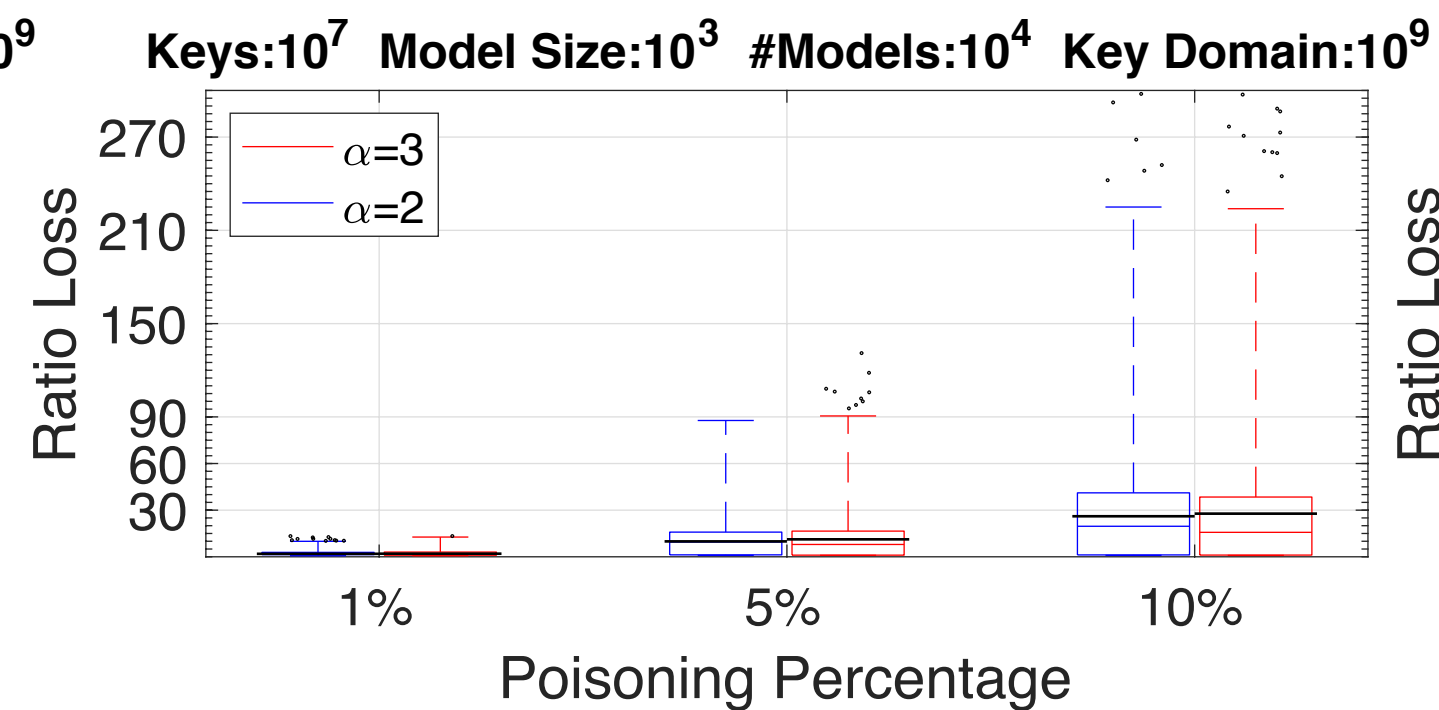


**150x larger MSE**

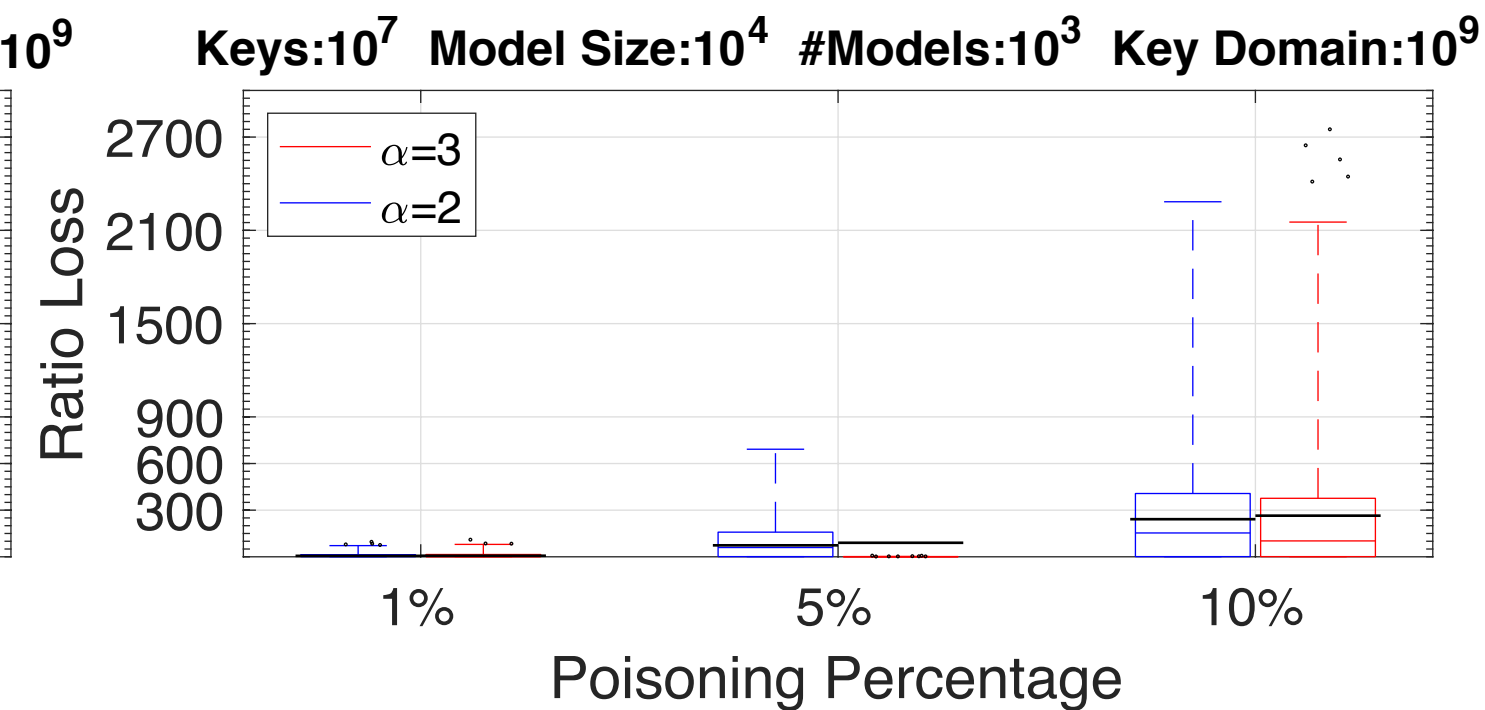
## Log-Normal Key Distribution



**3x larger MSE**



**30x larger MSE**



**300x larger MSE**



# CONCLUSION NEXT STEPS

## The Price of Tailoring the Index to Your Data: Poisoning Attacks on Learned Index Structures

Evgenios M. Kornaropoulos  
George Mason University, USA  
evgenios@gmu.edu

Silei Ren  
Cornell University, USA  
sr2262@cornell.edu

Roberto Tamassia  
Brown University, USA  
roberto@tamassia.net

### ABSTRACT

The concept of *learned index structures* relies on the idea that the input-output functionality of a database index can be viewed as a prediction task and, thus, implemented using a machine learning model instead of traditional algorithmic techniques. This novel angle for a decades-old problem has inspired exciting results at the intersection of machine learning and data structures. However, the advantage of learned index structures, i.e., the ability to adjust to the data at hand via the underlying ML model, can become a disadvantage from a security perspective as it could be exploited.

In this work, we present the first study of data poisoning attacks on learned index structures. Our poisoning approach is different from all previous works since the model under attack is trained on a cumulative distribution function (CDF) and, thus, every injection on the training set has a cascading impact on multiple data values. We formulate the first poisoning attacks on linear regression models trained on a CDF, which is a basic building block of the proposed learned index structures. We generalize our poisoning techniques to attack the advanced two-stage design of learned index structures called recursive model index (RMI), which has been shown to outperform traditional B-Trees. We evaluate our attacks under a variety of parametrizations of the model and show that the error of the RMI increases up to 340x and the error of its second-stage models increases up to 1000x.

### CCS CONCEPTS

• Information systems → Data structures; • Security and privacy → Cryptanalysis and other attacks; • Computing methodologies → Machine learning approaches.

### KEYWORDS

Learned Systems, Data Poisoning, Attacks, Indexing

### 1 INTRODUCTION

Database systems rely on *index structures* to access stored data efficiently. It is known to the database community that the motto “one size fits all” does not apply to traditional indexing schemes [24] since each index provides different performance guarantees that depend on the access pattern, the nature of the workload, and the underlying hardware. Even after choosing an appropriate index structure for a specific application, it is usually the case that a database administrator has to manually fine-tune the parameters of the system, either through experience or with help from tools. The work by Kraska, Beutel, Chi, Dea, and Polyzotis [30] challenged the state of affairs by re-framing index structures as a *machine learning* problem where the index directs a query to a memory location(s) based on a trained model tailored on the data at hand.

**Learned Index Structures.** The core idea of a *learned index structure* (LIS) is to model a data structure as a *prediction task*, i.e., get an input key and predict its location in a sorted sequence of key-record pairs. This approach allows the use of (i) continuous functions to encode the data, and (ii) *learning algorithms* to approximate the function. The specific LIS approach proposed by Kraska *et al.* [30] is to build the *cumulative distribution function* (CDF) for the keys. Given a key,  $k$ , the CDF returns the probability that a key chosen according to this distribution takes value less than or equal to  $k$ . Since the above probability is built from the set of keys at hand, it is expressed as the ratio of the number of keys less than  $k$  to the total number of keys. Given this insight, one can use the CDF to (i) compute the number of keys less than the (queried) key  $k$ , and (ii) infer the key’s memory location assuming the keys were sorted during the initialization. Therefore, a simple linear regression on the CDF gives an approximate location of the queried key. Indeed a linear regression on the CDF is one of the building blocks that has been shown to work well [30] and can be combined with *hierarchical models*, also called recursive model index (RMI) structures, so as to balance the final model for latency, memory usage, and computational cost. The hierarchy can be seen as building a mixture of “experts” [39] responsible for subsets of the data. The notion of a LIS has spurred a surge of works that blend ideas from machine learning, data structures, and systems (e.g., [5, 7, 10, 12–14, 17–21, 23, 24, 29, 31, 35, 36, 41, 43, 45, 47–49, 52, 55, 56, 58, 59]).

**First Vulnerability Assessment of Learned Index.** As promising as it may sound to combine ideas from machine learning and data structures, no analysis has been performed to understand potential vulnerabilities of the LIS paradigm. Intuitively, the advantage of a LIS is that the model adapts to the data at hand. However this efficiency might be problematic if the adversary is capable of injecting *maliciously crafted data* before the training of the model, i.e., at the initialization stage of the index structure, so as to cause inaccurate predictions of the location of legitimate data.

The technique of *data poisoning* has been known to be an effective attack vector for over a decade, e.g., see the references in [26]. In the context of static index structures, we focus on the case where the data stored in the index comes from multiple sources as different entities directly or indirectly contribute data, e.g., by generating data with their actions or behavior. A malicious actor can tailor its contributed data to deteriorate the index performance. Indeed, the real-world datasets used in the original LIS work [30] come from multiple contributors and, thus, are susceptible to poisoning attacks. Other examples of indexed data generated by multiple sources include data from personalized medicine, where patients voluntarily contribute their own data, as well as cybersecurity analytics where any user can submit its own indicators of compromise. Our threat model, much like all poisoning works [3, 4, 26, 60, 51], assumes that

● **First vulnerability assessment** for Learned Indexes

● Introduced the **“security mindset”** to discover blindspots on learned systems

● **Constructive dialog** between Database and Security communities

# Thank you!



<https://encrypted.systems>



[evgenios@gmu.edu](mailto:evgenios@gmu.edu)