1

Selectivity Estimation

Chuan Xiao

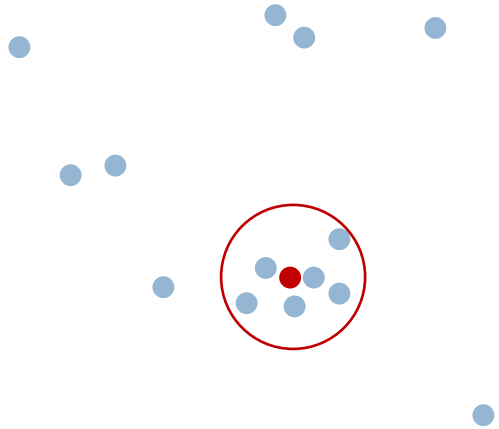
Osaka University and Nagoya University

Outline

- Problem Definition
- Applications
- Methods
- Performance Evaluation

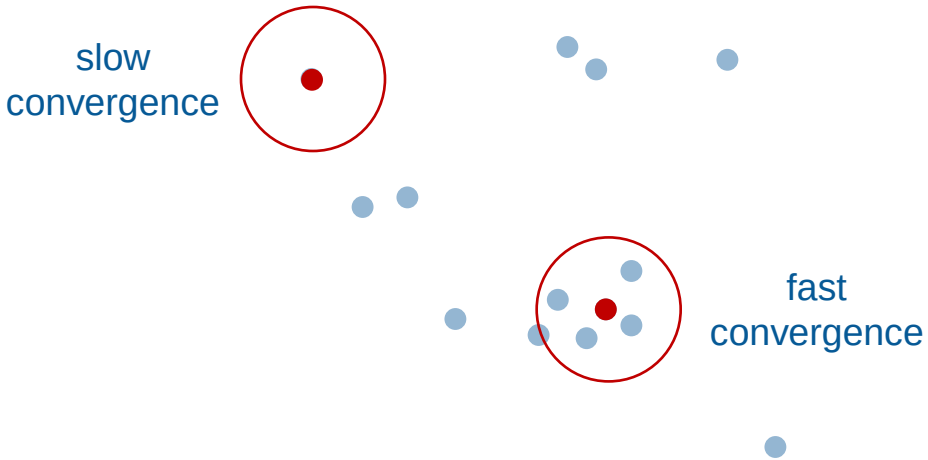
Problem Definition

- Selectivity estimation of similarity search for high-dimensional data
  - Given:
    - a database X of high-dimensional vectors,
    - a query vector $\mathbf{q}$,
    - a distance function $dist(.,.)$,
    - a threshold t.
  - Estimate the number of objects $\mathbf{x}$ in X such that $dist(\mathbf{q}, \mathbf{x}) \leq t$.
  - a.k.a. cardinality estimation, spherical range counting.
- Related problem
  - Selectivity (cardinality) estimation for relational data [KKRL+19, OBGK19, SL19, WSRY19, YLKW+19, HTAK+20, PZM20]
    - Each predicate deals with a dimension.
      - `SELECT COUNT(*) FROM employee WHERE age < 30 AND salary > 50000`
    - Dimensionality is usually low.



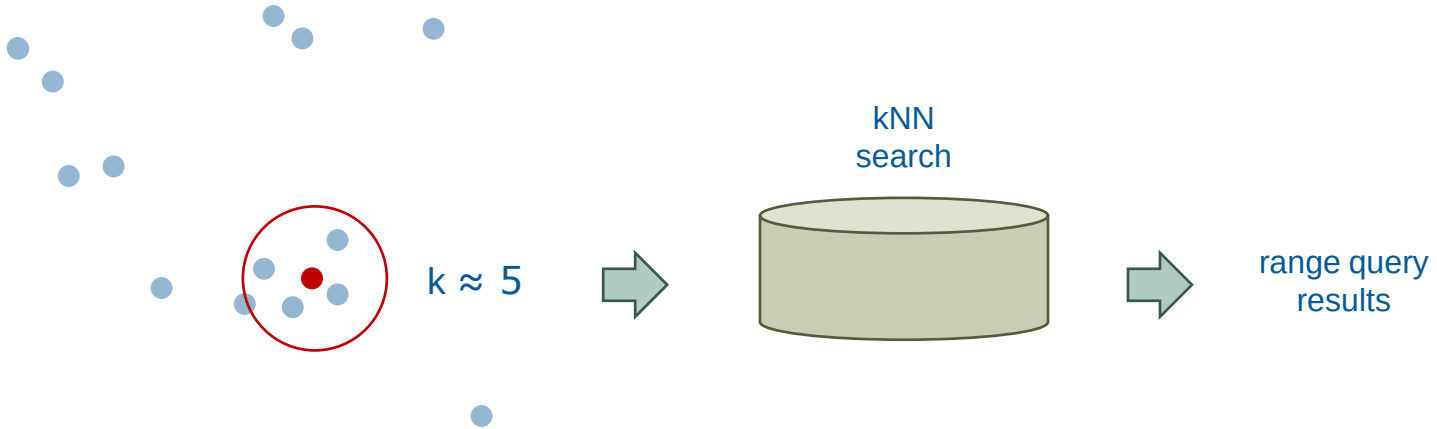
Application – Local Density Estimation

- Clustering
  - Find starting points.



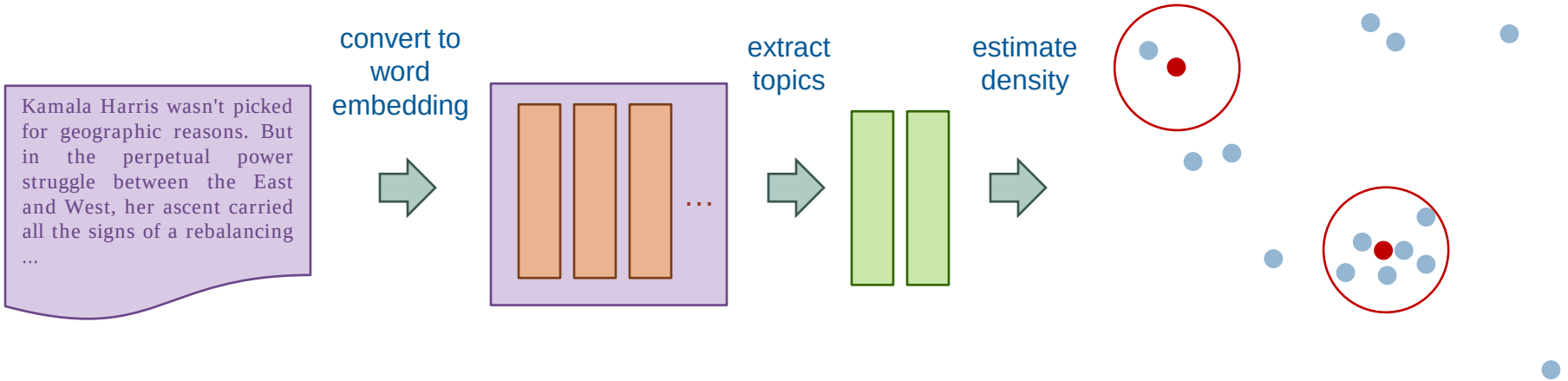
Application – Range Query Processing

- Convert range queries to kNN queries.
  - Some high-dimensional search algorithms are designed only for kNN queries (e.g., HNSW).
  - Estimate the number of results, and then apply a kNN algorithm.



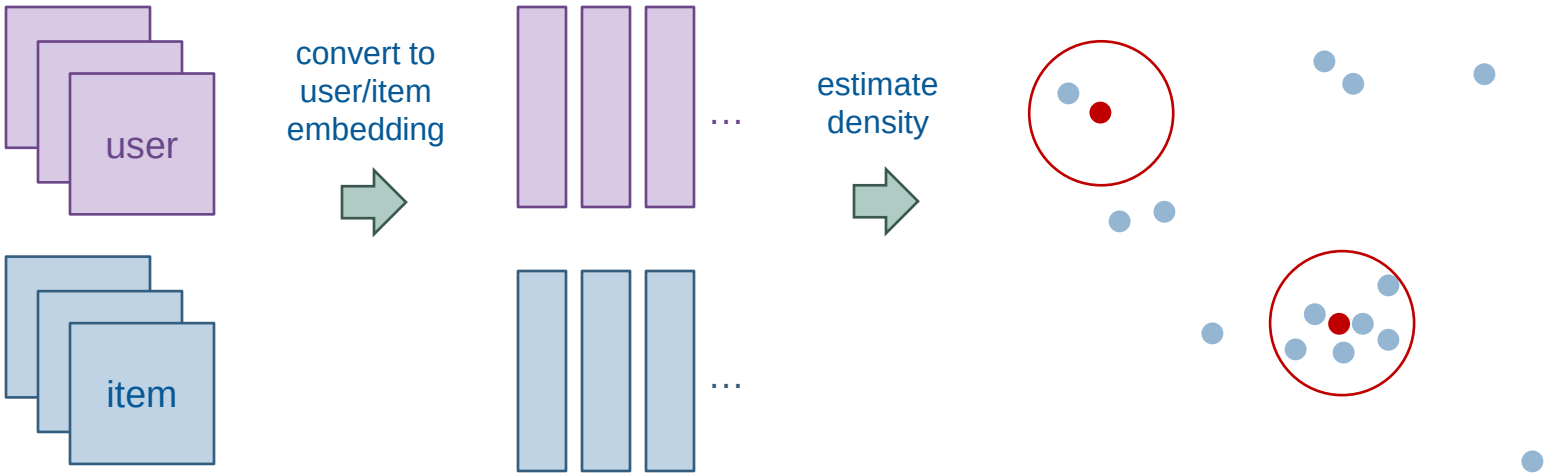
Application – Local Density Estimation

- Determine the popularities of topics.



Application – Local Density Estimation

- Find out if a user/item is an outlier in an e-commerce application.



Application – Image Retrieval

database image

to HD vector

1100010
1101101

index

postprocessing

output

…

query image

look up for candidates

1101011
1101001

source: Wikipedia

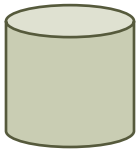
Application – Image Retrieval

database image

to HD vector

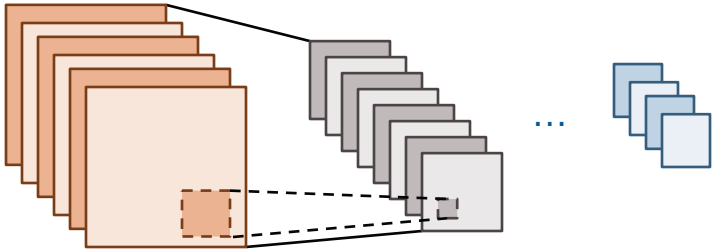
1100010
1101101

index



postprocessing

output



query image

look up for candidates

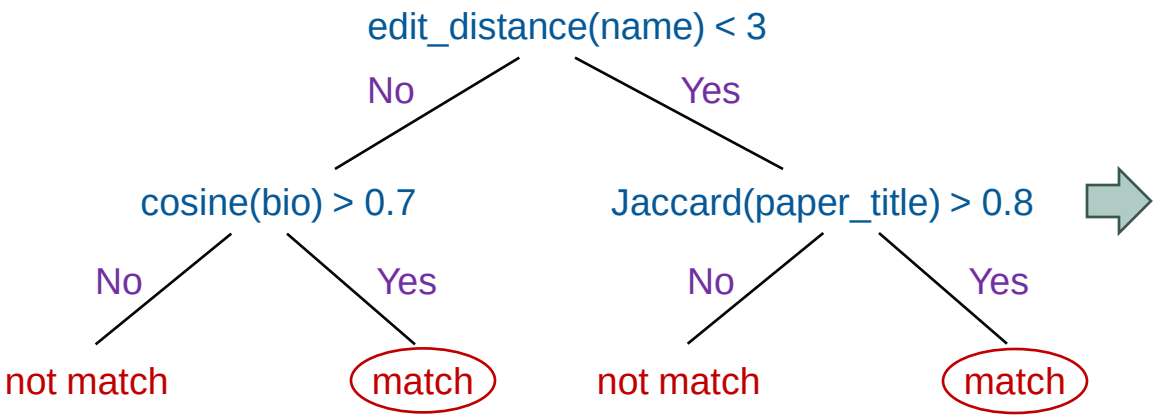
1101011
1101001

Estimate candidate size → running time, service level agreement …

source: Wikipedia

Application – Query Optimization

- Hands-off entity matching systems (e.g., Falcon [DCDN+17]) extract paths from random forests and take each path (a conjunction of similarity predicates) as a blocking rule.

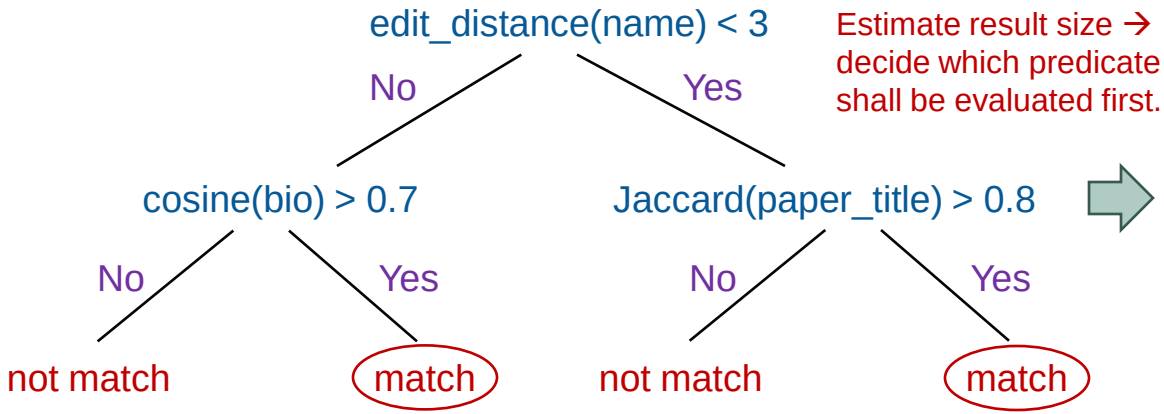


blocking rule 1
cosine(embedded_name) > 0.7
AND
cosine(embedded_bio) > 0.8

blocking rule 2
cosine(embedded_name) > 0.7
AND
cosine(embedded_paper_title) > 0.9

Application – Query Optimization

- Hands-off entity matching systems (e.g., Falcon [DCDN+17]) extract paths from random forests and take each path (a conjunction of similarity predicates) as a blocking rule.
- Embed textual attributes (e.g., by edit distance embedding [DYZW+20]) and process the conjunctive query.



Estimate result size → decide which predicate shall be evaluated first.

blocking rule 1
cosine(embedded_name) > 0.7
AND
cosine(embedded_bio) > 0.8

blocking rule 2
cosine(embedded_name) > 0.7
AND
cosine(embedded_paper_title) > 0.9

Evaluation Criteria of Selectivity Estimation

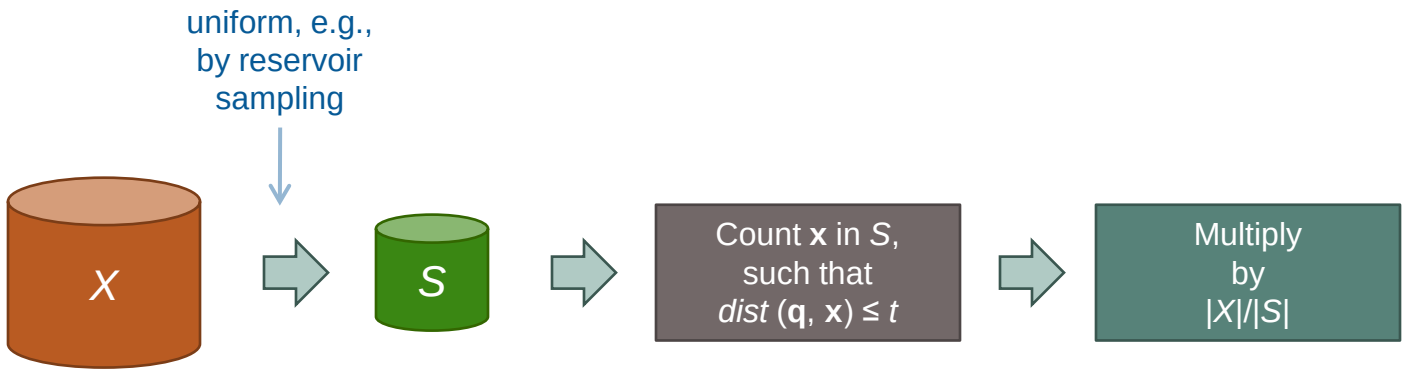
- Accuracy
  - Measured by MSE, MAPE, q-error, etc.
- Estimation speed
- Offline processing speed
  - Build an index?
  - Train a model?
- Performance guarantee
  - ε-δ
- Consistency (monotonicity)
  - For a fixed query object, selectivity is non-decreasing in the threshold.
  - This yields more interpretability and less vulnerability.
- Updatability
  - The database may have updates.

(Representative) Selectivity Estimation Methods

- Sampling
  - Uniform sampling
  - Importance sampling
- Kernel density estimation
- Regression
  - XGBoost
  - Vanilla deep neural network
  - Recursive model index
  - Deep lattice network
  - Threshold partitioning (CardNet)
  - Quantized regression
  - Query-dependent piecewise linear function (SelNet)
  - Global-local model

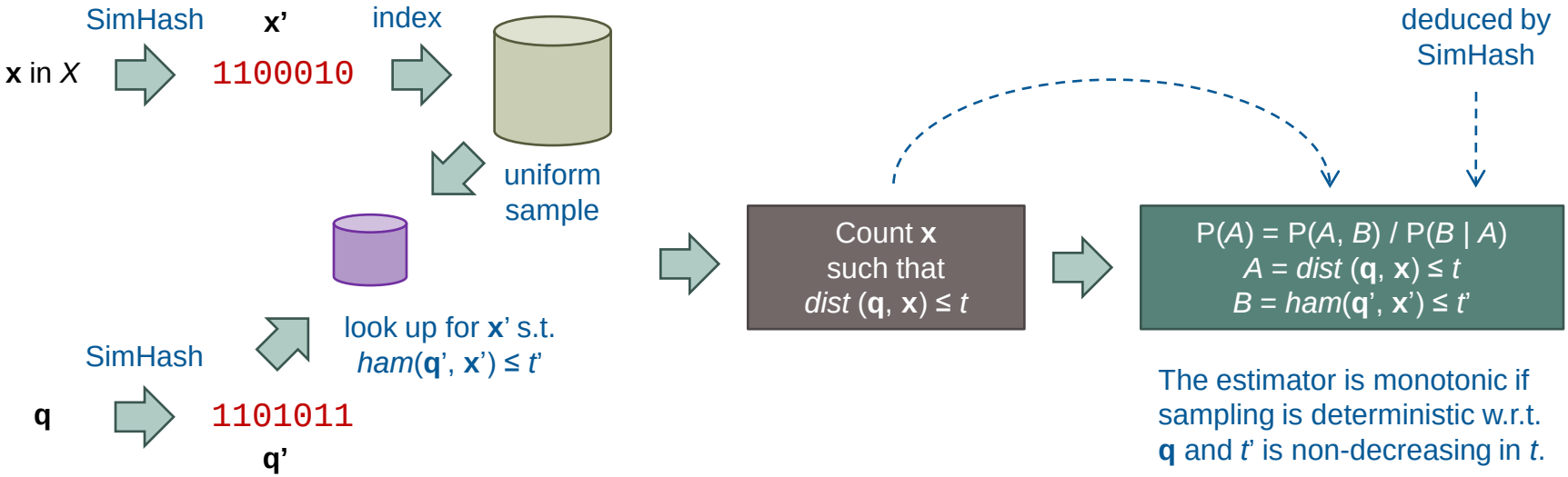
Sampling – Uniform Sampling

- A natural baseline
  - It is lightweight with well-understood performance, and easy to support monotonicity and handle updates.
  - Weakness: the probability that *dist* (**q**, **x**) ≤ *t* is small, especially when **q** is an outlier. So we need a very large sample size for accurate estimation.



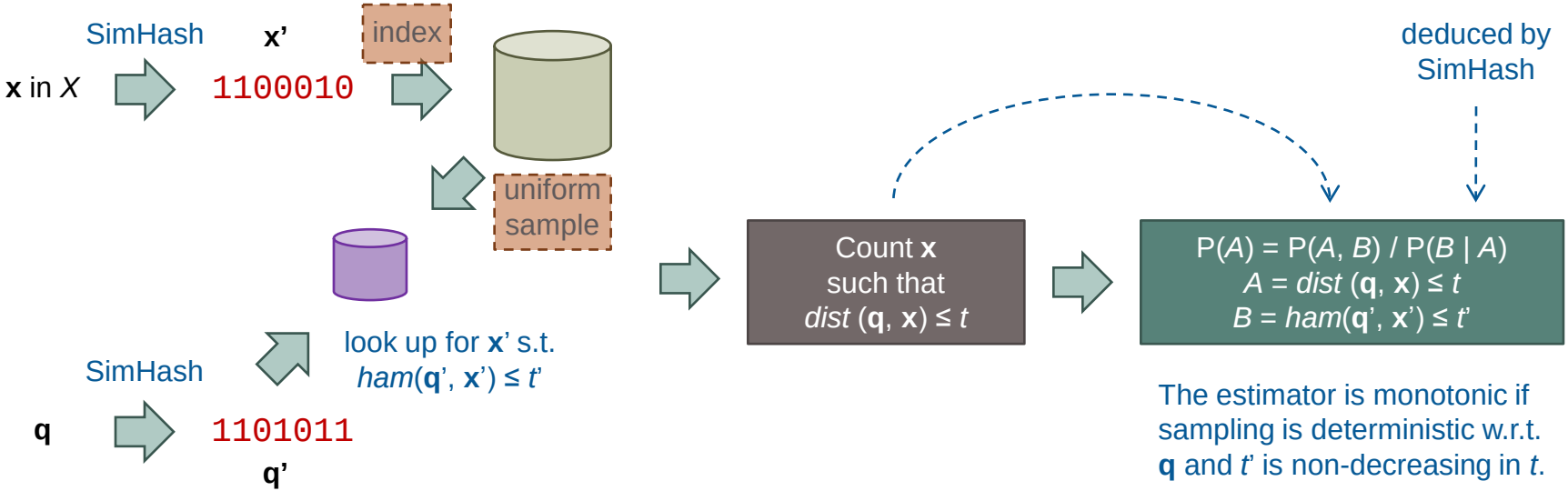
Sampling – Importance Sampling

- Estimate selectivity by generating samples from another distribution.
  - SimHash for angular distance (cosine similarity) [WCN18].
    - *dist*(., .) is captured by Hamming distance between hash values.
    - Use *L* independent hash tables for better accuracy.



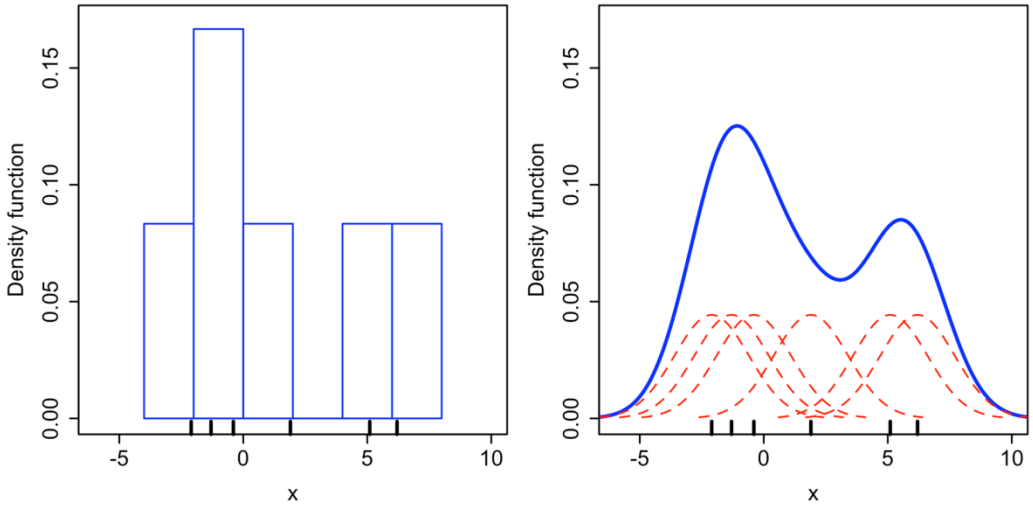
Sampling – Importance Sampling

- Deal with updates.
  - Update the index and sample.

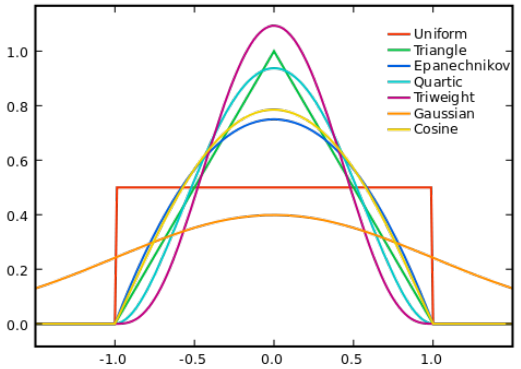


Kernel Density Estimation (KDE)

- Sample and smooth by kernel.



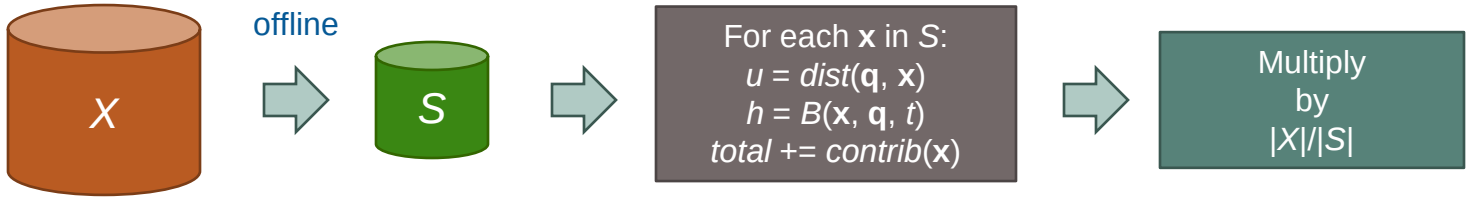
$$\widehat{f}_h(x) = \frac{1}{n}\sum_{i=1}^{n} K_h(x - x_i) = \frac{1}{nh}\sum_{i=1}^{n} K\left(\frac{x - x_i}{h}\right)$$

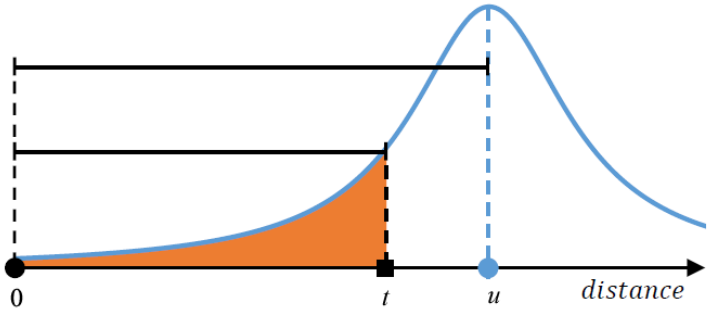


source: Wikipedia

Kernel Density Estimation (KDE)

- Model the probability density function of *dist*(**q**, **x**) by KDE [MFBS18].
  - Sample objects and compute their contributions to the selectivity.





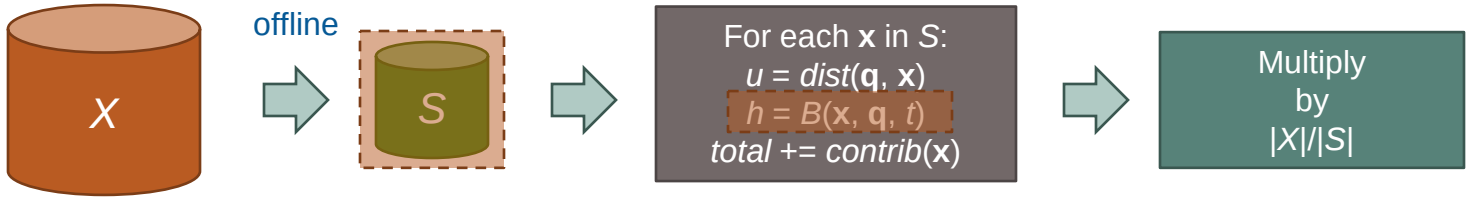
source: [MFBS18]

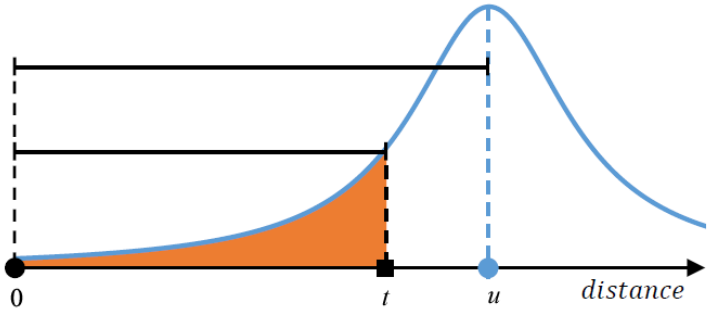
B(): a bandwidth function learned using query samples. The estimator is monotonic if *B*() is independent of *t*.

$$contrib(\mathbf{x}) = \int_0^t K_h(x-u)dx$$

Kernel Density Estimation (KDE)

- Deal with updates.
  - Incrementally sample more objects.
  - Retrain bandwidth functions.





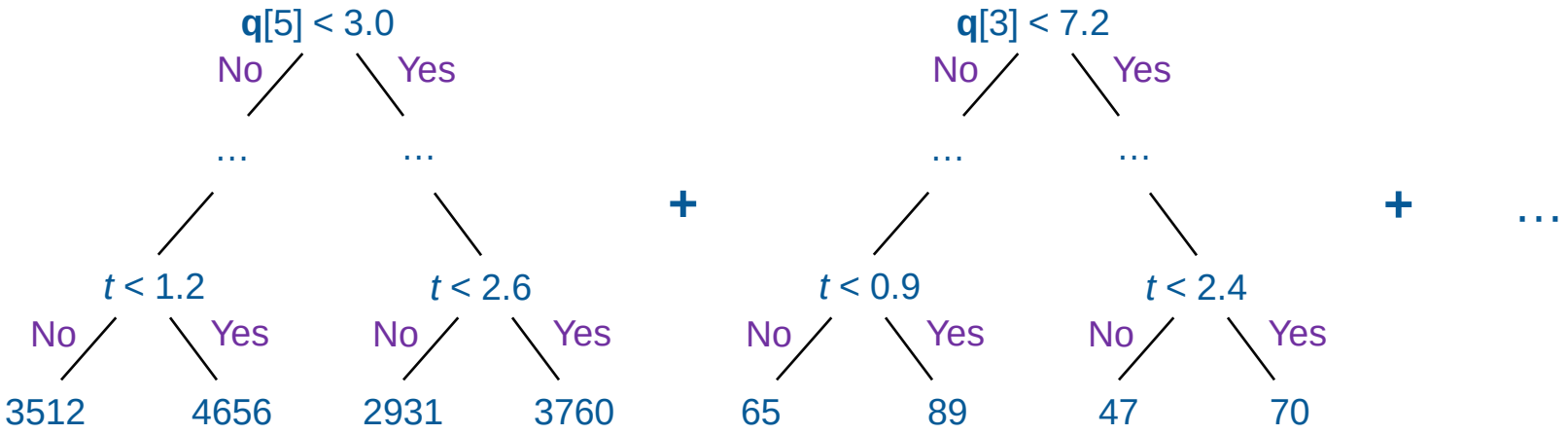
source: [MFBS18]

B(): a bandwidth function learned using query samples. The estimator is monotonic if *B*() is independent of *t*.

$$contrib(\mathbf{x}) = \int_0^t K_h(x-u)dx$$

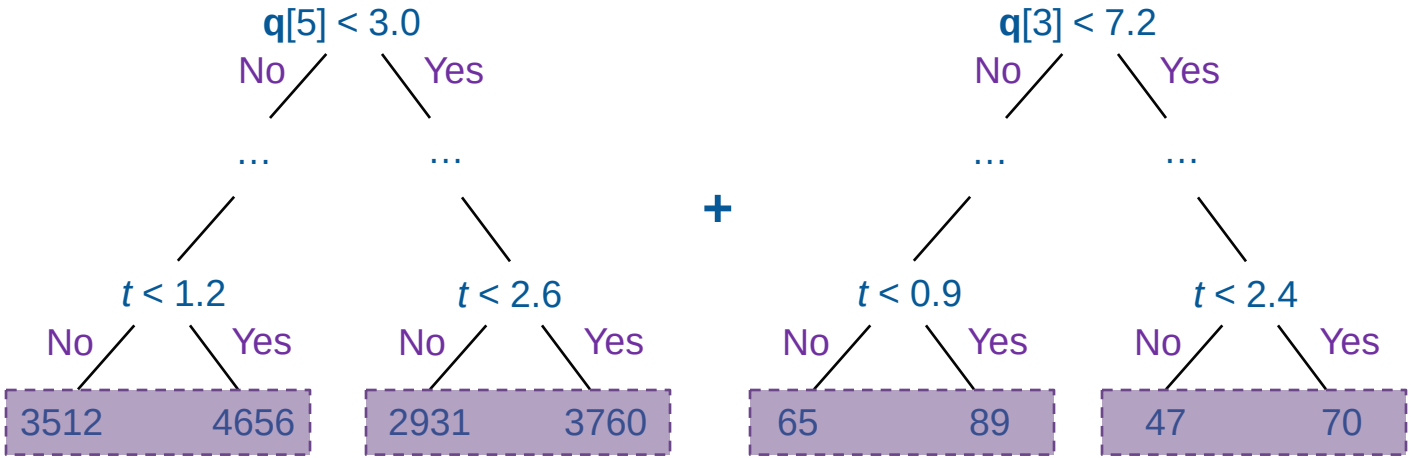
Regression – XGBoost

- Gradient boosting [CG16]
  - Ensemble of weak prediction models.
    - For example, decision trees, with each rule in the form of **q**[i] < α or *t* < β.
  - Each model is learned to fit the residual of previous ones.



Regression – XGBoost

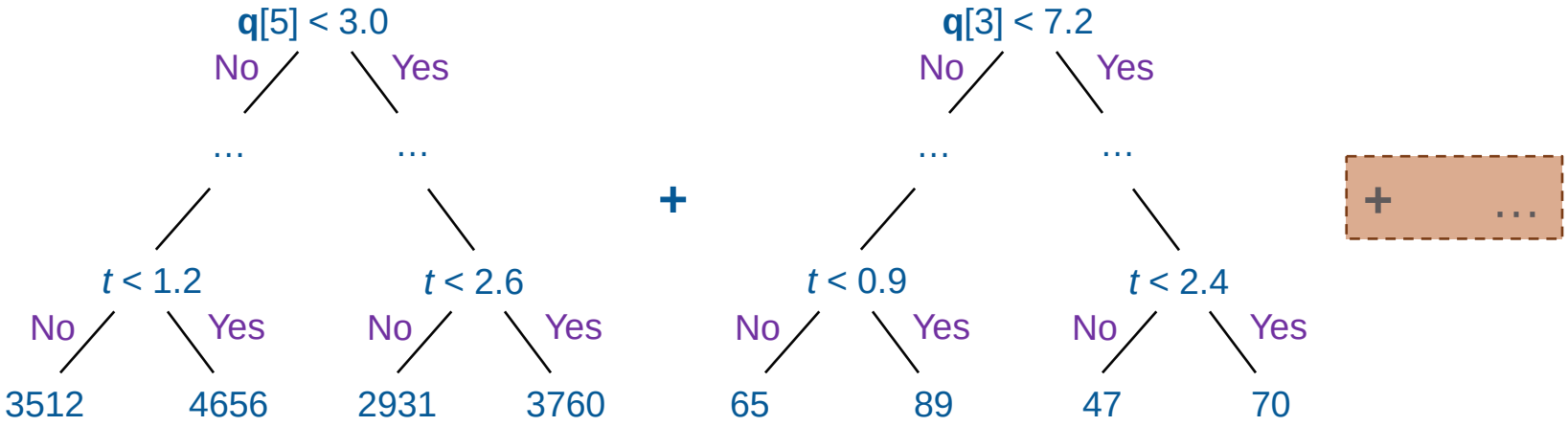
- Gradient boosting [CG16]
  - Ensemble of weak prediction models.
    - For example, decision trees, with each rule in the form of **q**[i] < α or *t* < β.
  - Each model is learned to fit the residual of previous ones.



The estimator is monotonic if *t* < β is only at the bottom level and leaf node values are non-decreasing w.r.t. Yes/No.

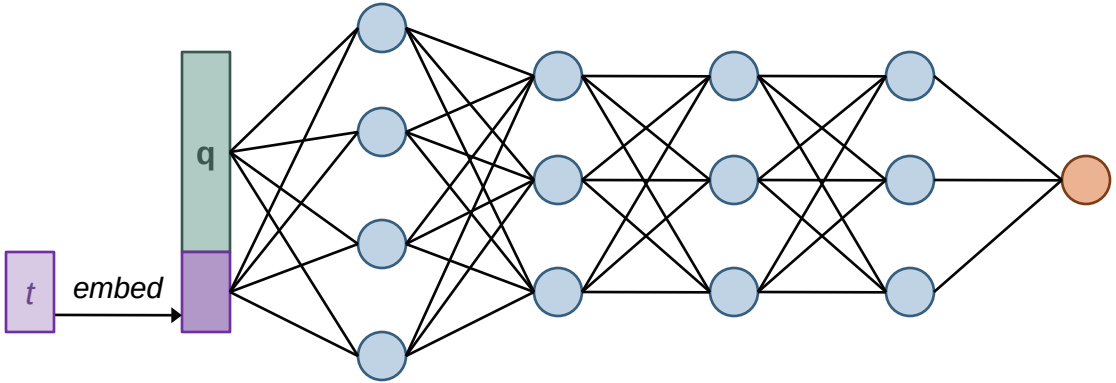
Regression – XGBoost

- Deal with updates.
  - It is time-consuming to retrain existing decision trees.
  - Train more decision trees to fit the residual.



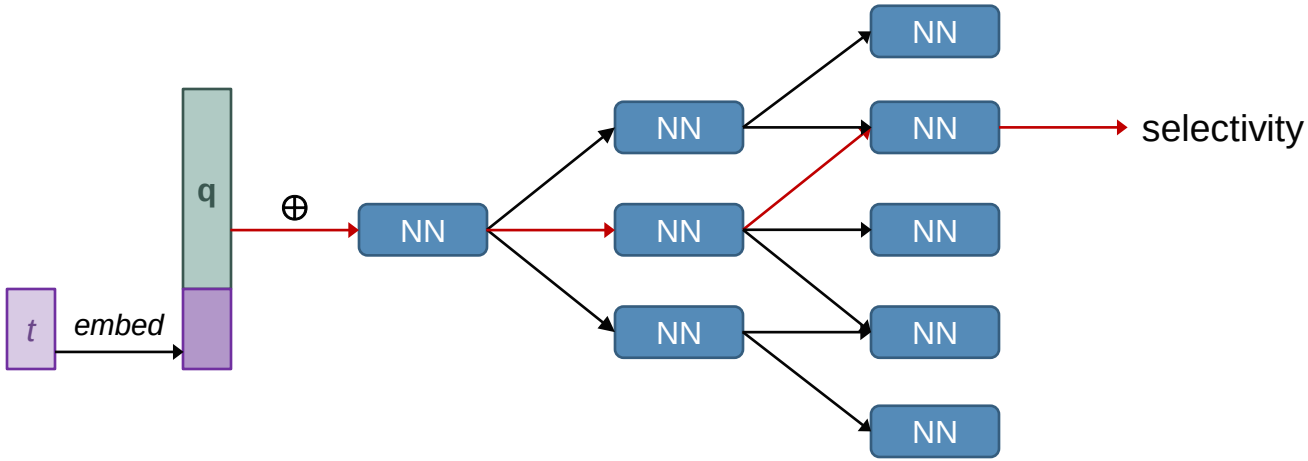
Regression – Vanilla Deep Neural Network

- Fully connected neural network.
  - Number of hidden layers ≈ 4.
  - For higher accuracy, embed *t* (dim ≈ 5) and concatenate to **q** as input.
  - Non-monotonic.



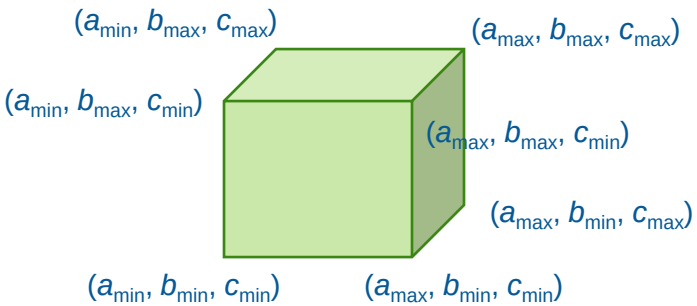
Regression – Recursive Model Index

- Originally developed for range indexing in relational databases [KBCD+18].
  - Inspired by the mixture-of-experts model.
  - Each model (e.g., a neural network) picks another one in the next stage.
  - Non-monotonic.



Regression – Deep Lattice Network

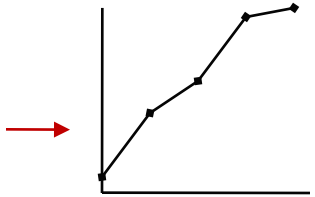
- Developed for monotonic regressions [YDCP+17].
  - Input: monotonic features + non-monotonic features.
- Components
  - Lattice: regression for a *d*-dimensional input.



Only learn for vertex values. Others are processed by multilinear interpolation.

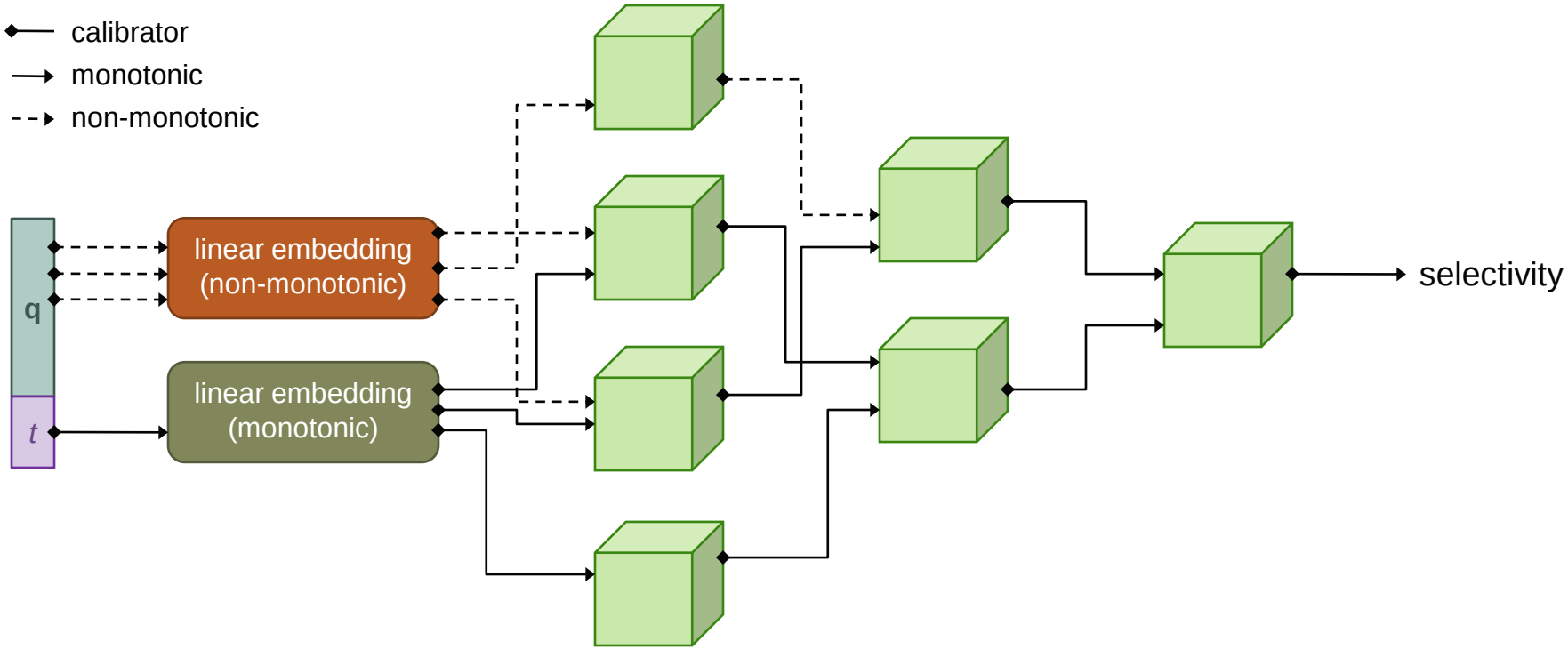
$lat(a, b, c) \le lat(a', b, c)$ for monotonic feature $a \le a'$.

- Calibrator: 1-dimensional non-decreasing piecewise linear function.



- Linear embedding: a matrix (all elements ≥ 0 for monotonicity).

Regression – Deep Lattice Network



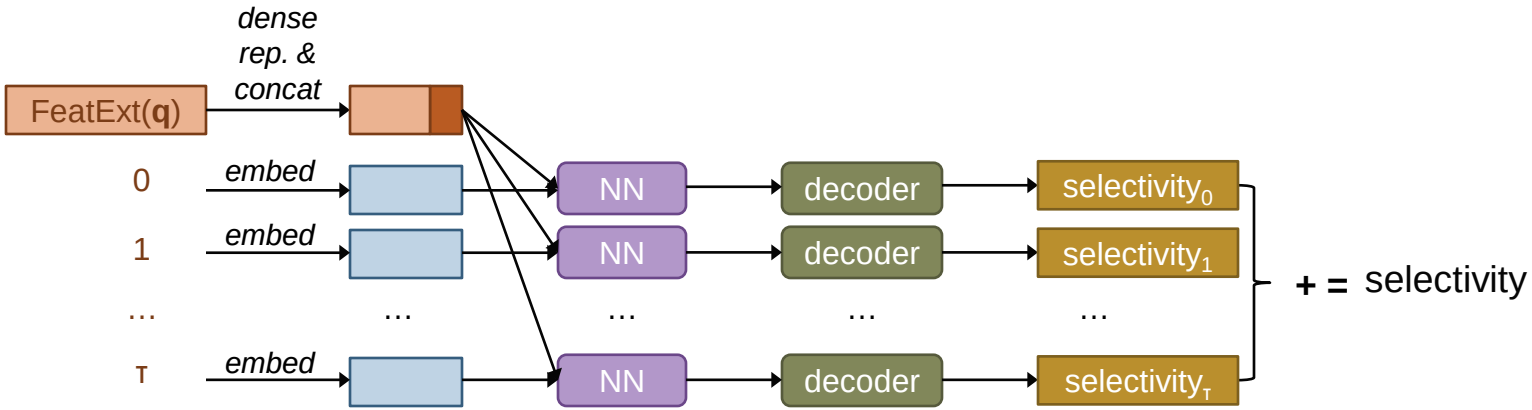
Regression – Threshold Partitioning (CardNet)

- Idea: partition the distance threshold, and then use multiple regressors, each dealing with a distance interval [WXQC+20].
- Procedure: feature extraction + regression
  - Feature extraction
    - query vector **q** → binary vector **r**
      - Map **q** to an integer *B* by LSH (e.g., random projection) and then set the *B*-th bit to 1
      - Repeat *L* rounds using *L* hash functions and concatenate the resulting bit vectors.

    - threshold t → integer τ
      - 0 → 0. t_{max} → τ_{max}. Other values are mapped in a non-decreasing manner → monotonicity.

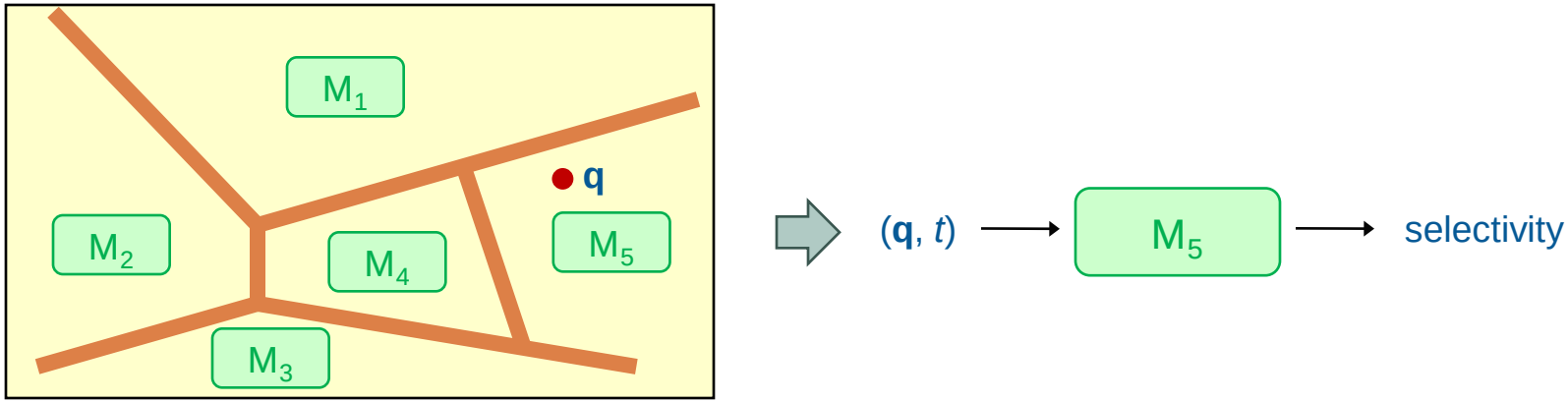
Regression – Threshold Partitioning (CardNet)

- Idea: partition the distance threshold, and then use multiple regressors, each dealing with a distance interval.
- Procedure: feature extraction + regression
  - Regression
    - Use (τ + 1) regressors, each for a distance in 0, 1, … τ.



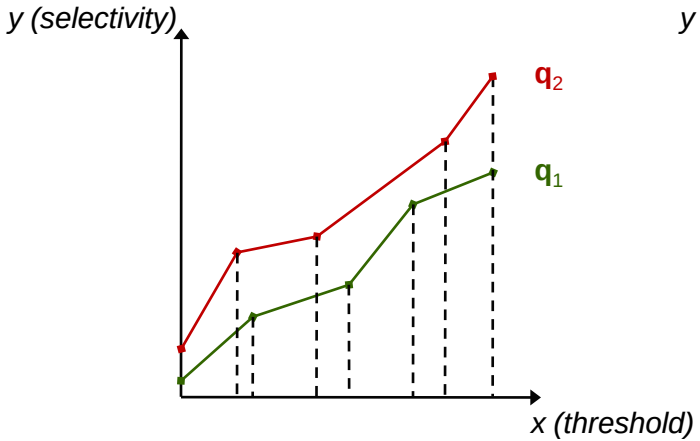
Regression – Quantized Regression

- Query space partitioning [AT15, AT17].
  - Quantize the query space to discover prototypes of query patterns.
  - Train a local model (e.g., a linear model, which is monotonic) to deal with each prototype.

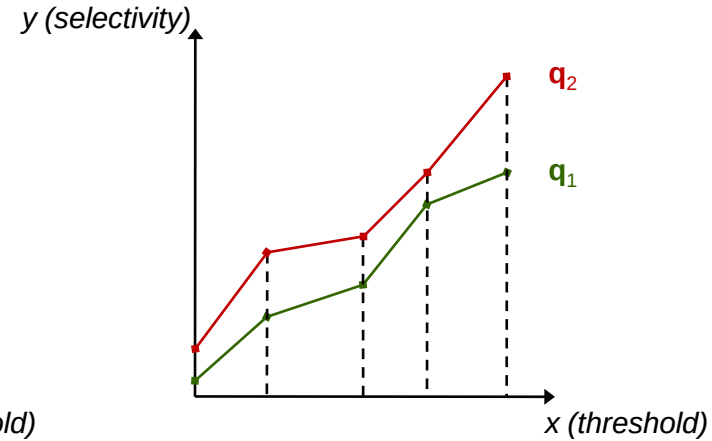


Regression – Query-Dependent Piecewise Linear Function (SelNet)

- Learn query patterns implicitly (in contrast to quantized regression) and utilize piecewise linear functions (PLFs) for regression [WXQM+20].
- Query-dependent PLFs v.s. query-independent PLFs.



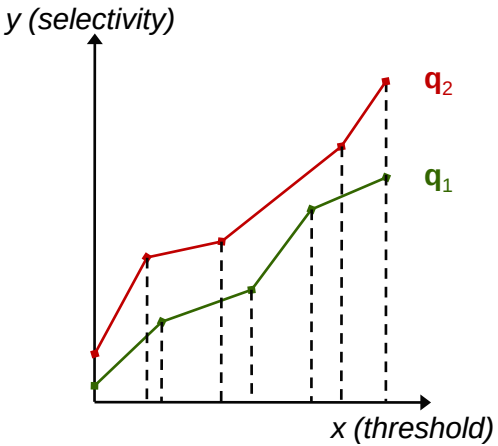
query-dependent:
variable (adaptive)
control points

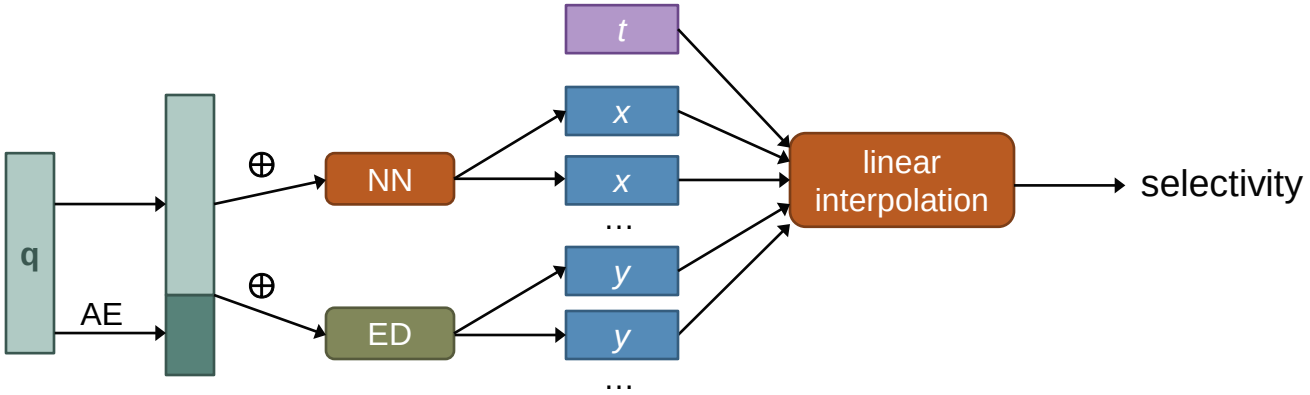


query-independent
(deep lattice network):
fixed control points

Regression – Query-Dependent Piecewise Linear Function (SelNet)

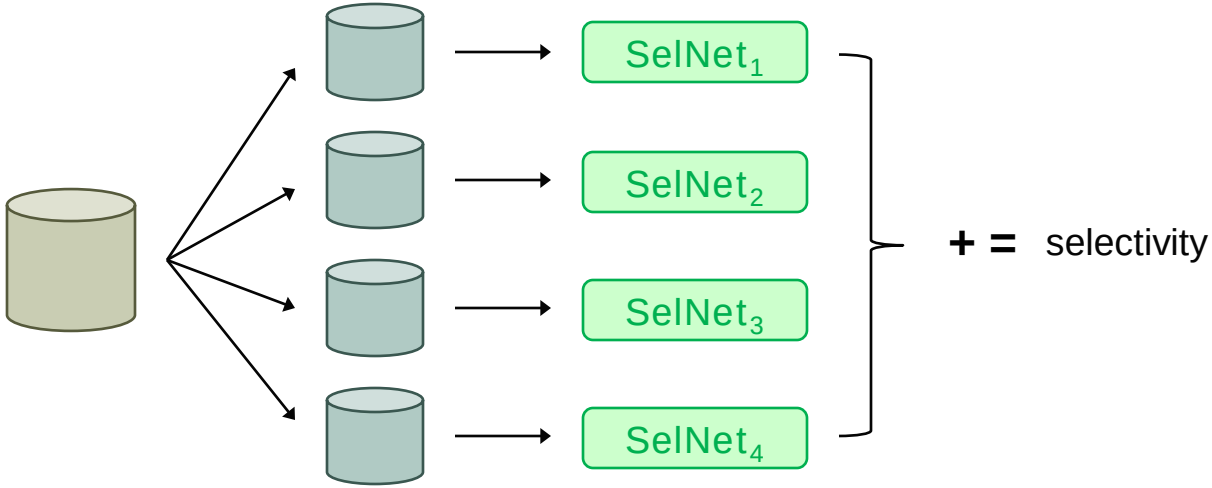
- Learn query-dependent PLFs.
  - NN: a neural network that outputs the *x* values of control points.
  - ED: an encoder-decoder model that outputs the *y* values of control points.
  - Monotonic if *y* is non-decreasing in *x*.





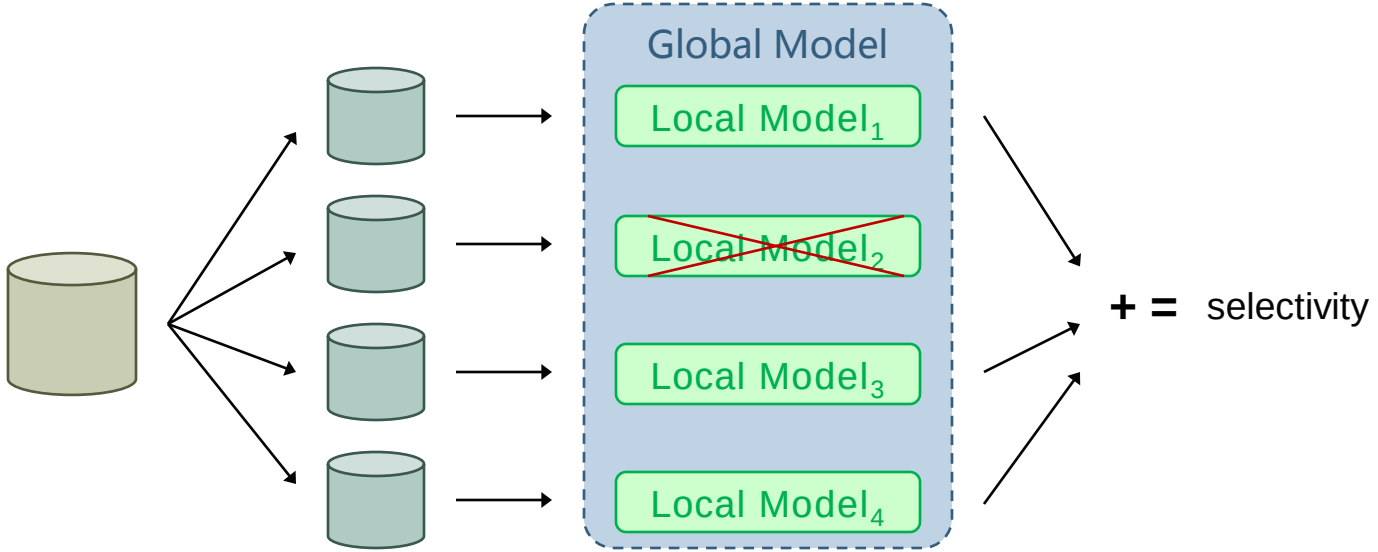
Regression – Query-Dependent Piecewise Linear Function (SelNet)

- Dataset partitioning
  - Density may vary across different regions in a large dataset.
  - Partition the dataset into n non-overlapping parts and train a local model for each part.
  - Sum up the selectivity estimated by each local model.



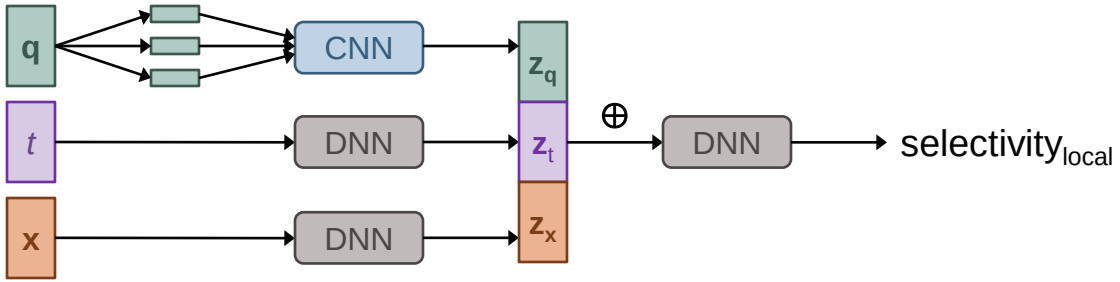
Regression – Global-Local Model

- Another dataset partitioning approach [SGT21].
  - Partition the dataset into non-overlapping parts and train a local model for each part.
  - Train a global model to select local models that produce non-zero selectivities.



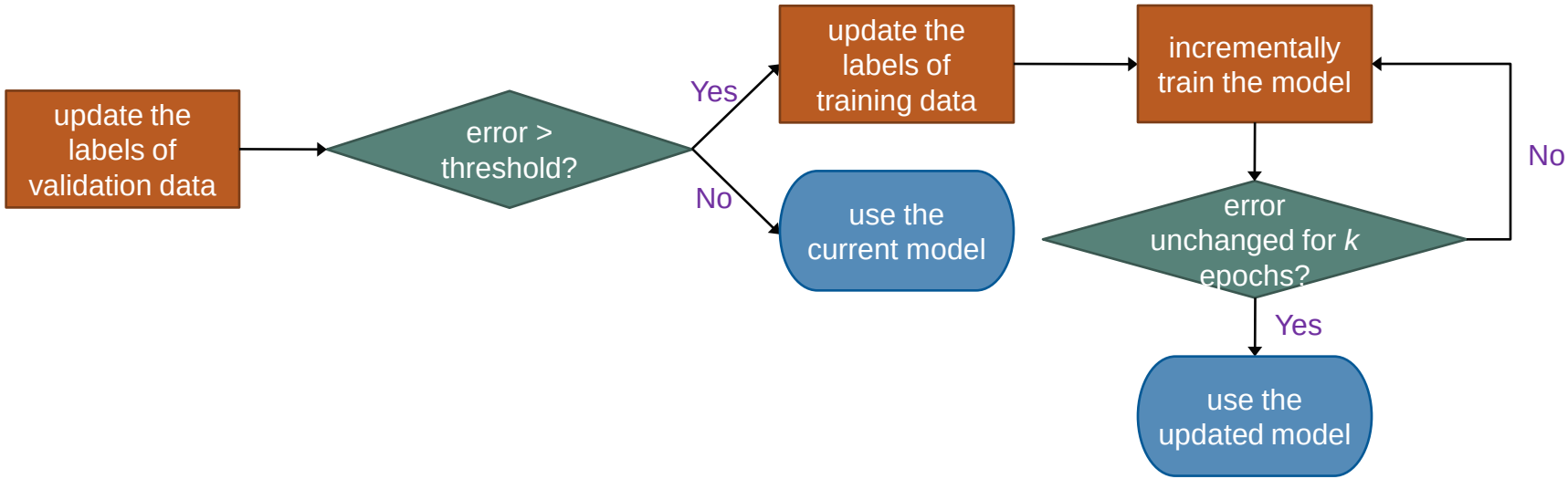
Regression – Global-Local Model

- Local model
  - Input
    - Query feature: **q** partitioned into *m* disjoint subvectors.
    - Threshold feature: a scalar t.
    - Dataset feature: a *k*-dimensional vector **x** such that each dimension is the similarity between **q** and a sample in the subset of dataset for this local model.
  - Procedure
    - Embed query, threshold, and dataset feature vectors respectively.
    - Concatenate the embeddings and feed to a feedforward neural network for regression.
    - Enforce all the weights in the threshold embedding to be positive to guarantee monotonicity.



Regression – Dealing with Updates

- Many (deep) regression models are trained through gradient descent.
- We adopt incremental learning for these models.



Benchmarks

- So far there are no specific benchmarks for this problem.
- Datasets used in existing work (benchmarks for other uses)
  - Text
    - GloVe: 1.9M 300-dimensional word embedding
      - https://nlp.stanford.edu/projects/glove/
    - fastText: 1M 300-dimensional word embedding
      - https://fasttext.cc/docs/en/english-vectors.html
  - Image
    - MS-Celeb-1M: 10M celebrity images for face recognition
      - https://msceleb.org/ (terminated in 2019)
      - Pre-processed by faceNet [SKP15] to 128-dimensional vectors.
  - Video
    - YouTube: 3.4K videos of 1.6K people
      - http://www.cs.tau.ac.il/~wolf/ytfaces/index.html
      - 0.35M 1770-dimensional (-feature) vectors extracted from the frames.

Comparison of Selectivity Estimation Methods

Method	Accuracy	Estimation Speed	Offline Proc. Speed	Performance Guarantee	Consistency (Monotonicity)
Uniform Sampling	Adjustable / Very Low	Adjustable	**None**	**Yes**	Possible
Importance Sampling	Adjustable / Low	Adjustable	**Fast**	**Yes**	Possible
KDE	Medium	Slow	Medium	No	Possible
XGBoost	Low	Medium	Medium	No	Possible
Vanilla DNN	Low	**Fast**	Medium	No	No
Recursive Model Index	Medium	Medium	Slow	No	No
DLN	Low	Slow	Slow	No	**Yes**
CardNet	**High**	**Fast**	Slow	No	**Yes**
Quantized Regression	Medium	Slow	Medium	No	Possible
SelNet	**High**	Medium	Slow	No	**Yes**
Global-Local	**High**	**Fast**	Slow	No	Possible

References

- C. Anagnostopoulos and P. Triantafillou. Learning set cardinality in distance nearest neighbours. In ICDM, pages 691-696, 2015.
- C. Anagnostopoulos and P. Triantafillou. Query-driven learning for predictive analytics of data subspace cardinality. ACM Trans. Knowl. Discov. Data, 11(4):47:1-47:46, 2017.
- T. Chen and C. Guestrin. XGBoost: A scalable tree boosting system. In KDD, pages 785–794, 2016.
- X. Dai, X. Yan, K. Zhou, Y. Wang, H. Yang, and J. Cheng. Convolutional embedding for edit distance. In SIGIR, pages 599–608, 2020.
- S. Das, P. S. G. C., A. Doan, J. F. Naughton, G. Krishnan, R. Deep, E. Arcaute, V. Raghavendra, and Y. Park. Falcon: Scaling up hands-off crowdsourced entity matching to build cloud services. In SIGMOD, pages 1431–1446, 2017.
- S. Hasan, S. Thirumuruganathan, J. Augustine, N. Koudas, and G. Das. Deep learning models for selectivity estimation of multi-attribute queries. In SIGMOD, pages 1035–1050, 2020.
- A. Kipf, T. Kipf, B. Radke, V. Leis, P. A. Boncz, and A. Kemper. Learned cardinalities: Estimating correlated joins with deep learning. In CIDR, 2019.
- T. Kraska, A. Beutel, E. H. Chi, J. Dean, and N. Polyzotis. The case for learned index structures. In SIGMOD, pages 489–504, 2018.
- M. Mattig, T. Fober, C. Beilschmidt, and B. Seeger. Kernel-based cardinality estimation on metric data. In EDBT, pages 349–360, 2018.
- J. Ortiz, M. Balazinska, J. Gehrke, and S. S. Keerthi. An empirical analysis of deep learning for cardinality estimation. arXiv preprint arXiv:1905.06425, 2019.

References

- Y. Park, S. Zhong, and B. Mozafari. Quicksel: Quick selectivity learning with mixture models. In SIGMOD, pages 1017–1033, 2020.
- N. Reimers and I. Gurevych. Sentence-BERT: Sentence embeddings using siamese bert-networks. In EMNLP-IJCNLP, pages 3980–3990, 2019.
- F. Schroff, D. Kalenichenko, and J. Philbin. Facenet: A unified embedding for face recognition and clustering. In CVPR, pages 815–823, 2015.
- N. Shazeer, A. Mirhoseini, K. Maziarz, A. Davis, Q. Le, G. Hinton, and J. Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. arXiv preprint arXiv:1701.06538, 2017.
- J. Sun and G. Li. An end-to-end learning-based cost estimator. PVLDB, 13(3):307–319, 2019.
- J. Sun, G. Li, and N. Tang. Learned cardinality estimation for similarity queries. In SIGMOD, pages 1745-1757, 2021.
- B. Walenz, S. Sintos, S. Roy, and J. Yang. Learning to sample: Counting with complex queries. PVLDB, 13(3):390–402, 2019.
- Y. Wang, C. Xiao, J. Qin, X. Cao, Y. Sun, W. Wang, and M. Onizuka. Monotonic cardinality estimation of similarity selection: A deep learning approach. In SIGMOD, pages 1197–1212, 2020.
- Y. Wang, C. Xiao, J. Qin, R. Mao, M. Onizuka, W. Wang, R. Zhang. Consistent and flexible selectivity estimation for high-dimensional data. arXiv preprint arXiv:2005.09908 (2020).
- A. Wehenkel and G. Louppe. Unconstrained monotonic neural networks. In NeurIPS, pages 1543–1553, 2019.
- X. Wu, M. Charikar, and V. Natchu. Local density estimation in high dimensions. In ICML, pages 5293–5301, 2018.
- Z. Yang, E. Liang, A. Kamsetty, C. Wu, Y. Duan, P. Chen, P. Abbeel, J. M. Hellerstein, S. Krishnan, and I. Stoica. Deep unsupervised cardinality estimation. PVLDB, 13(3):279–292, 2019.
- S. You, D. Ding, K. Canini, J. Pfeifer, and M. Gupta. Deep lattice networks and partial monotonic functions. In NIPS, pages 2981–2989, 2017.

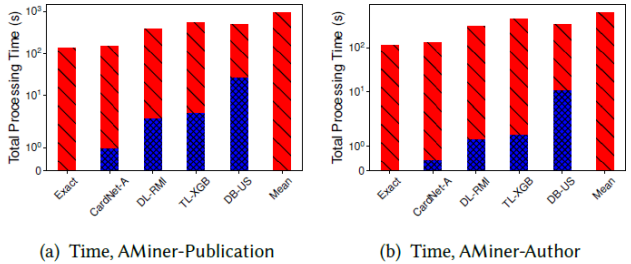
Performance in a Query Optimizer

- Datasets
  - AMiner (author names & publications)
  - IMDB (cast & movie titles)
  - Attributes are pre-processed by Sentence-BERT [RG19].
- Queries
  - Conjunctive queries of 2 – 5 Euclidean distance predicates.
    - For example, *dist*(name) ≤ 0.25 AND *dist*(affiliations) ≤ 0.4 AND *dist*(research interests) ≤ 0.45.
  - For each query, we estimate the selectivity of each predicate.
  - The predicate with the smallest selectivity is evaluated first by index lookup. Others are checked on the fly.
- Methods
  - Uniform sampling
  - XGBoost
  - RMI
  - CardNet-A: CardNet with acceleration for estimation
  - Exact: an oracle that instantly returns the true selectivity.
  - Mean: an estimator that returns the same selectivity (mean of 10,000 random queries) for a given threshold.

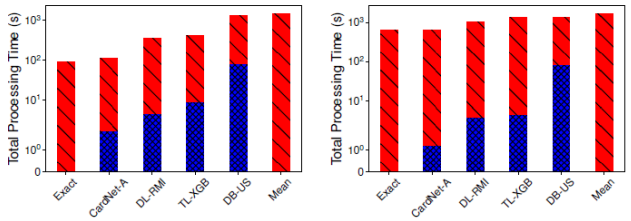
Performance in a Query Optimizer

- Exact: oracle (true selectivity)
- Cardnet-A
- DL-RMI: RMI
- TL-XGB: XGBoost
- DB-US: uniform sampling
- Mean: same (mean selectivity)

Query processing time:
estimation / lookup+check

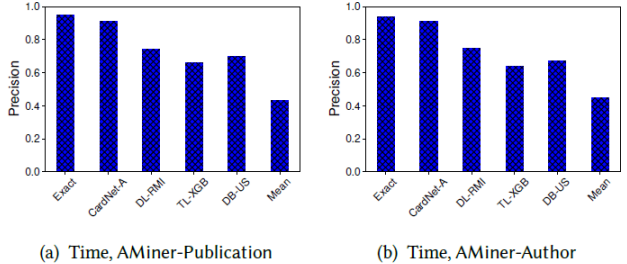


(a) Time, AMiner-Publication (b) Time, AMiner-Author

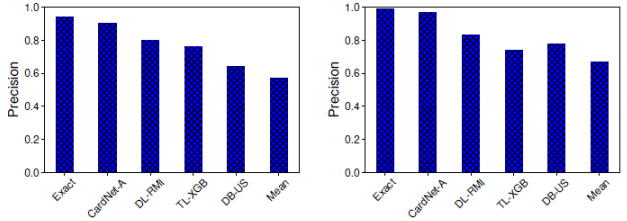


(c) Time, IMDB-Movie (d) Time, IMDB-Actor

Precision of query planning:
% of queries on which a method picks the fastest plan



(a) Time, AMiner-Publication (b) Time, AMiner-Author



(c) Time, IMDB-Movie (d) Time, IMDB-Actor