# Selectivity Estimation

Chuan Xiao

Osaka University and Nagoya University

# Outline

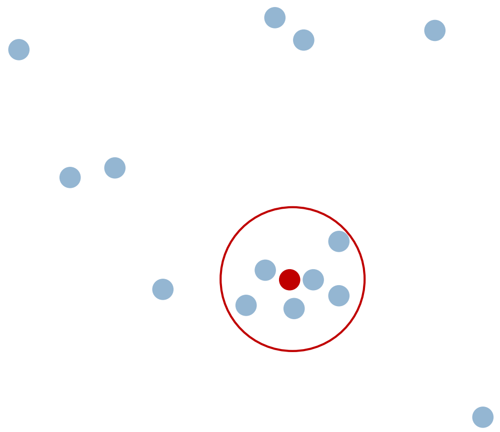


- Problem Definition
- Applications
- Methods
- Performance Evaluation

# Problem Definition



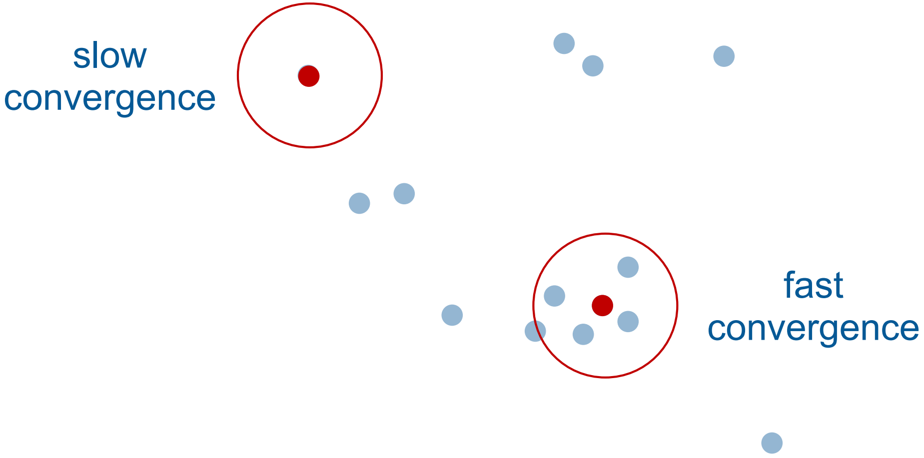
- Selectivity estimation of similarity search for high-dimensional data
  - Given:
    - a database $X$ of high-dimensional vectors,
    - a query vector $\mathbf{q}$,
    - a distance function $dist(.,.)$,
    - a threshold $t$.
  - Estimate the number of objects $\mathbf{x}$ in $X$ such that $dist(\mathbf{q}, \mathbf{x}) \le t$.
  - 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



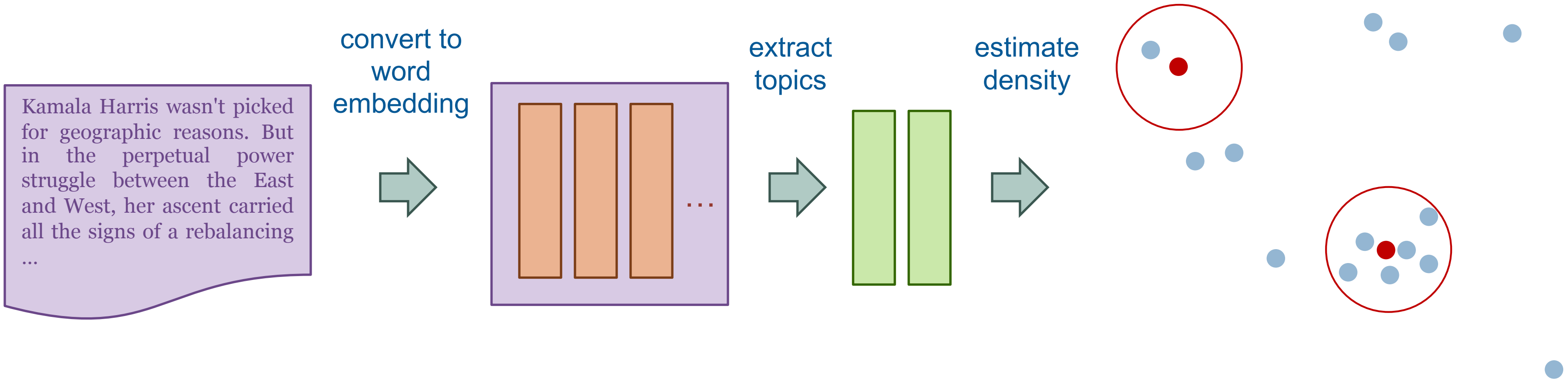
- Clustering
  - Find starting points.



# Application – Local Density Estimation



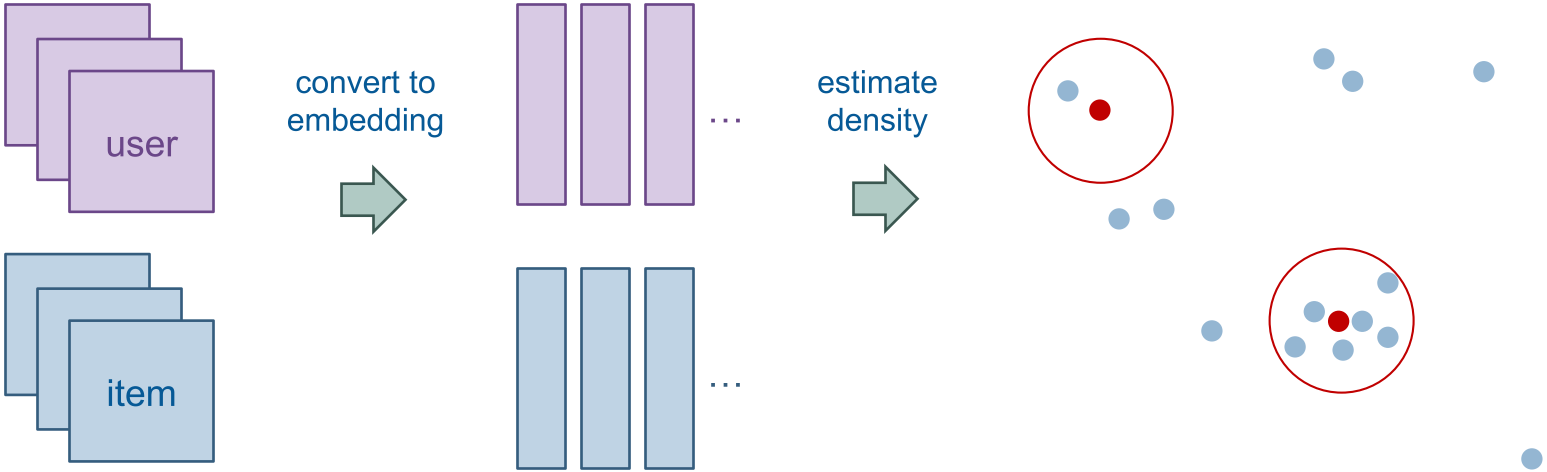
- Determine the popularities of topics.



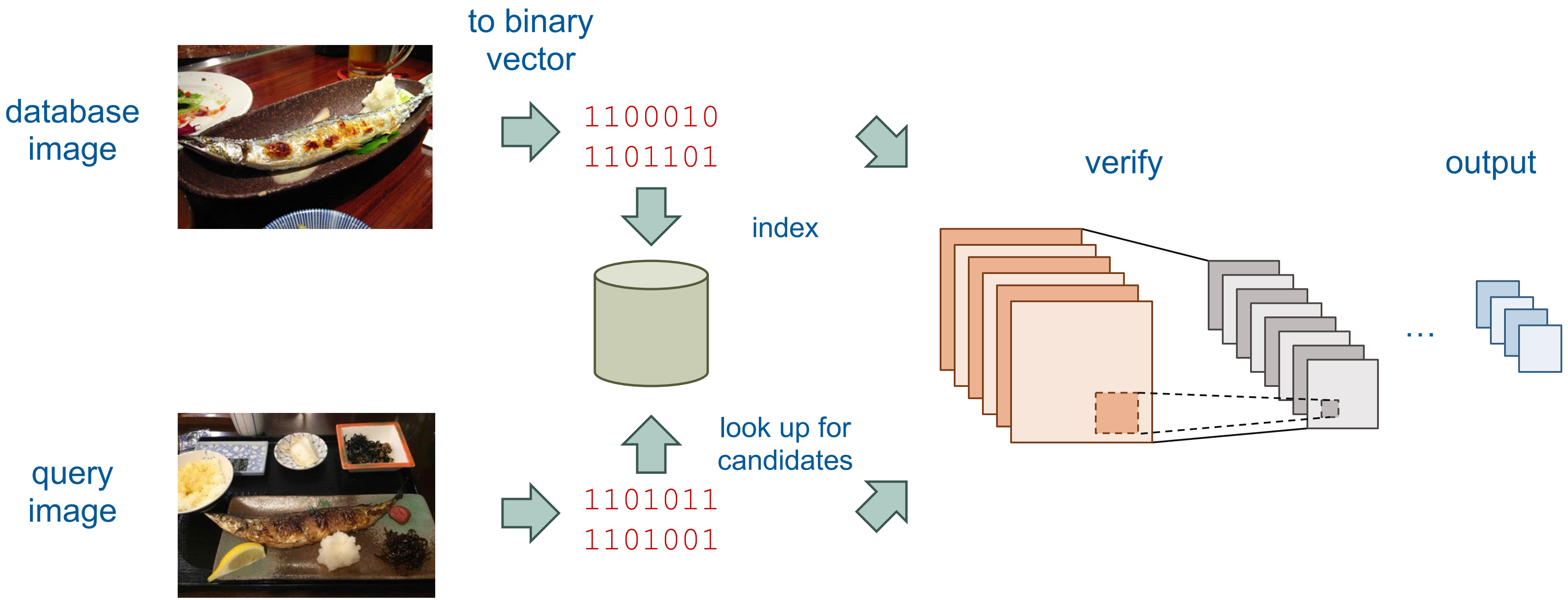
# Application – Local Density Estimation



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



# Application – Image Retrieval




source: Wikipedia

# Application – Image Retrieval

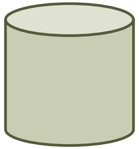


database image

to binary vector

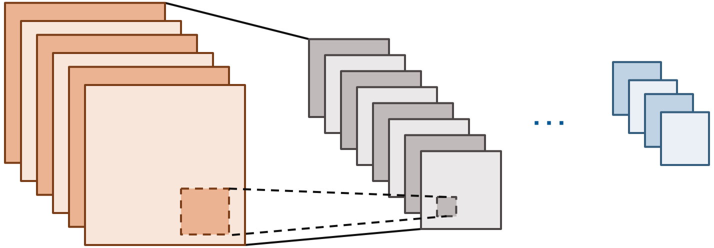
1100010
1101101

index



verify

output



look up for candidates

query image

1101011
1101001

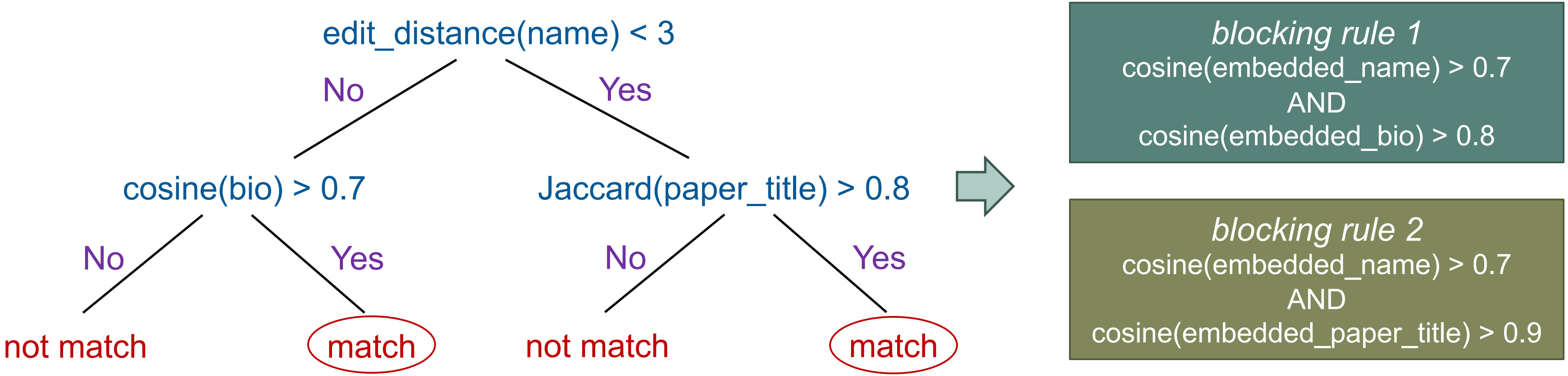
source: Wikipedia

Estimate candidate size
→ running time, SLA …

# Application – Query Optimization



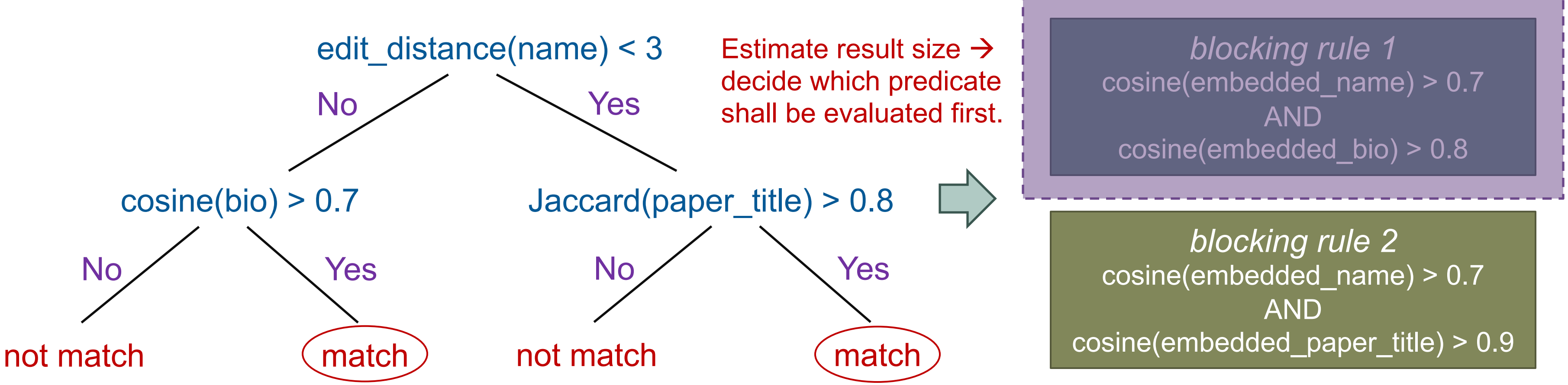
- Hands-off entity matching systems (e.g., Falcon [DCDN+17]) extract paths from random forests and take each path (a conjunction of similarity predicates) as a blocking rule.



# 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.



# 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
  - $\varepsilon$-$\delta$
- Consistency (monotonicity)
  - For a fixed query object, selectivity is <u>non-decreasing</u> in the threshold.
  - This yields more interpretability and less vulnerability.
- Updatability
  - The database may have updates.

# (Representative) Selectivity Estimation Methods

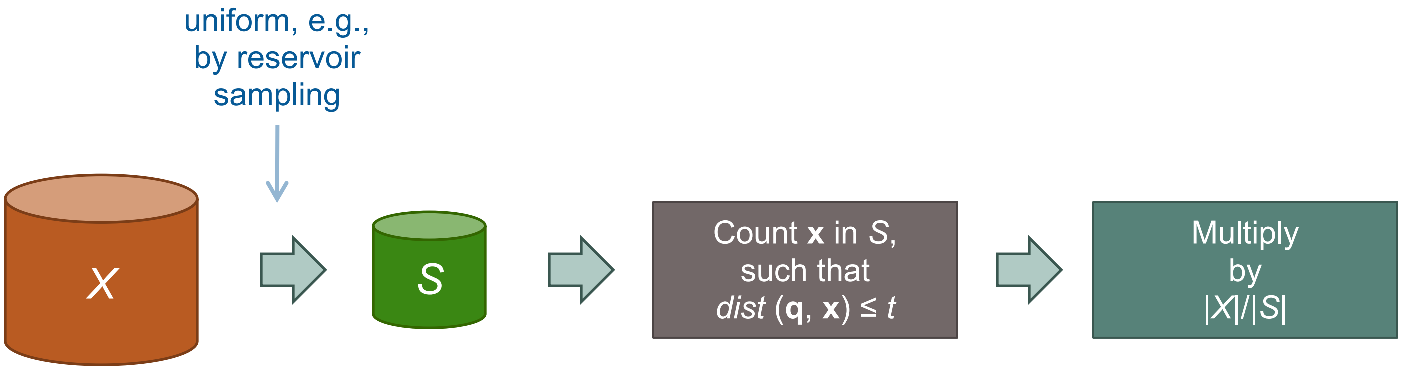


- Sampling
- Kernel density estimation
- Regression
  - Non-deep learning
    - XGBoost
  - Deep learning
    - Vanilla deep neural network
    - Recursive model index
    - Deep lattice network
    - CardNet (incremental prediction + deep learning)
    - SelNet (piecewise linear function + deep learning)

# Sampling – Uniform Sampling



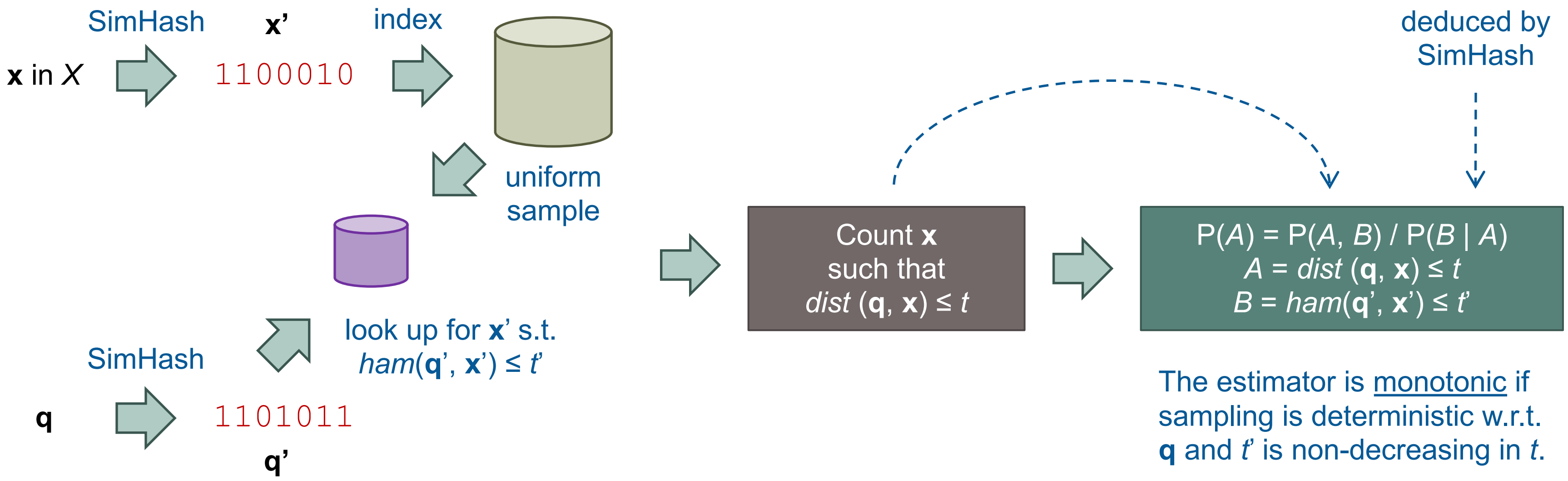
- A natural baseline
  - It is lightweight with well-understood performance, and easy to support monotonicity and handle updates.
  - Weakness: the probability that *dist* (**q**, **x**) ≤ *t* is small, especially when **q** is an outlier. So we need a very large sample size for accurate estimation.



# Sampling – Importance Sampling



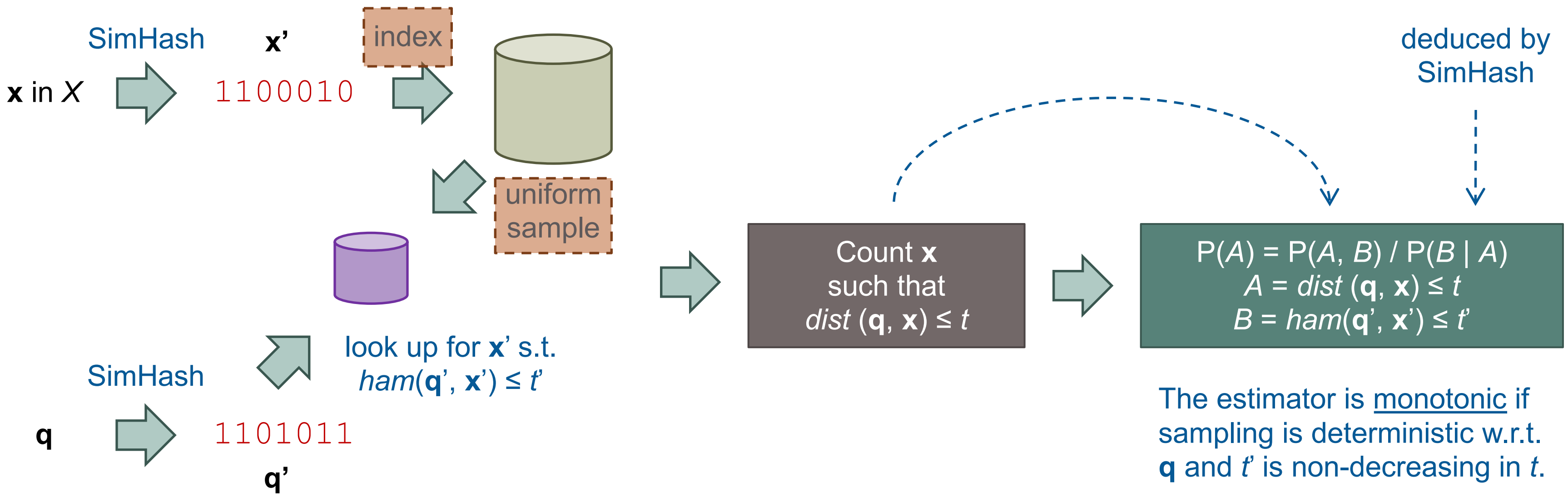
- Estimate selectivity by generating samples from another distribution.
  - SimHash for angular distance (cosine similarity) [WCN18].
    - *dist*(., .) is captured by Hamming distance between hash values.
    - Use *L* independent hash tables for better accuracy.



# Sampling – Importance Sampling



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

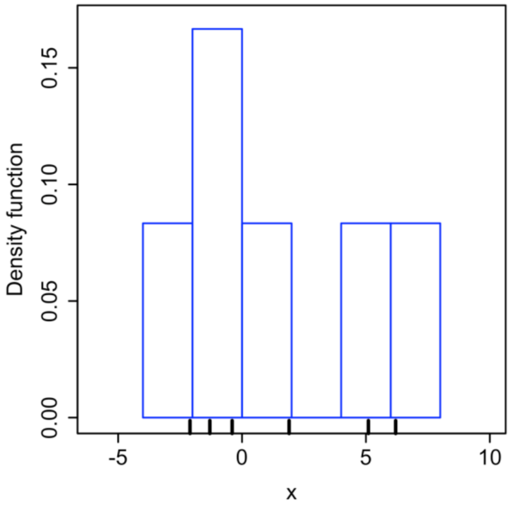


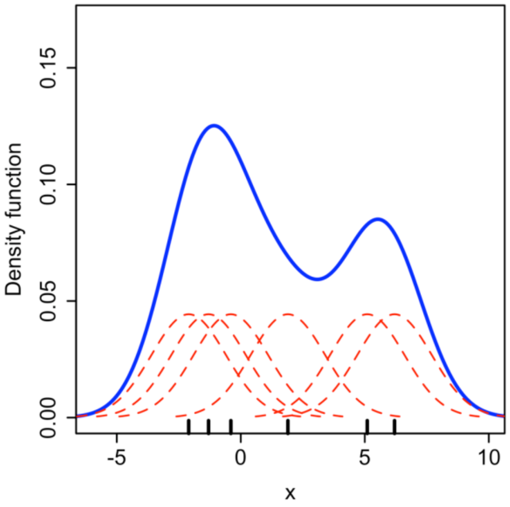

The estimator is monotonic if sampling is deterministic w.r.t. **q** and *t'* is non-decreasing in *t*.

# Kernel Density Estimation (KDE)

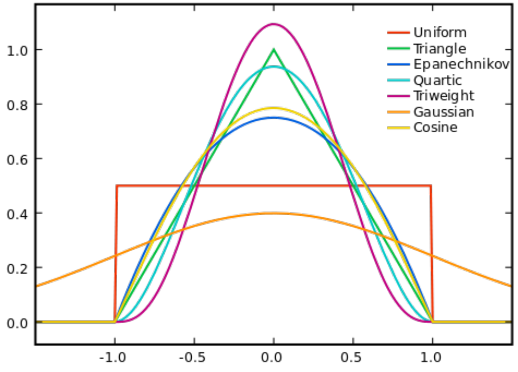


- Sample and smooth by kernel




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

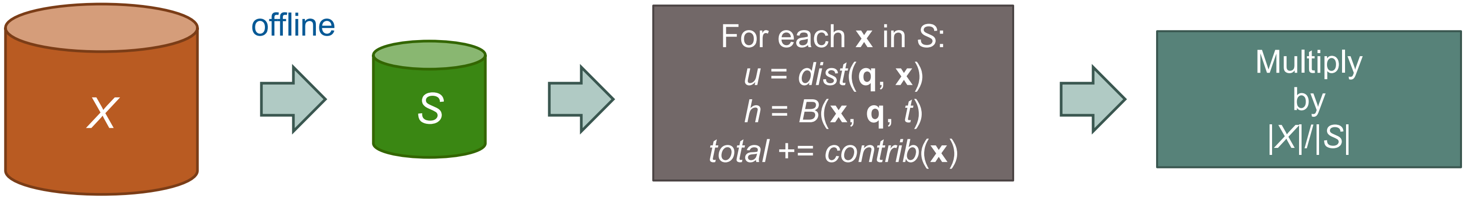


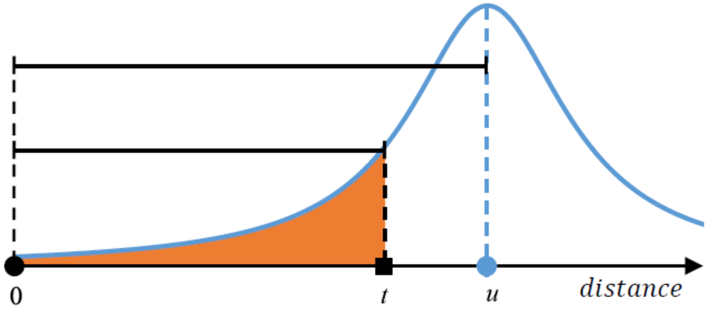

source: Wikipedia

# 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]

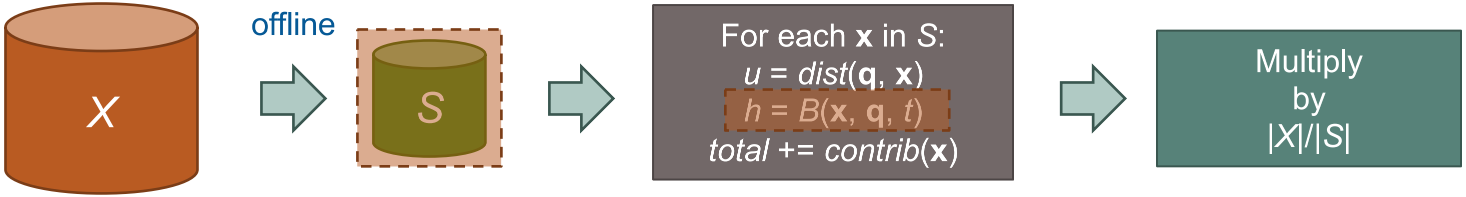
*B*(): a bandwidth function learned using query samples. The estimator is <u>monotonic</u> if *B*() is independent of *t*.

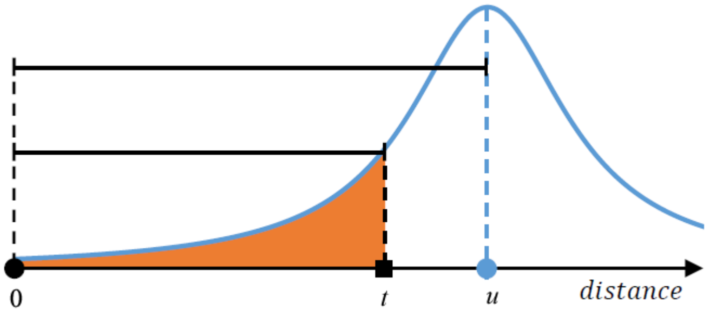
$$contrib(\mathbf{x}) = \int_0^t K_h(x-u)dx$$

# Kernel Density Estimation (KDE)



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




source: [MFBS18]

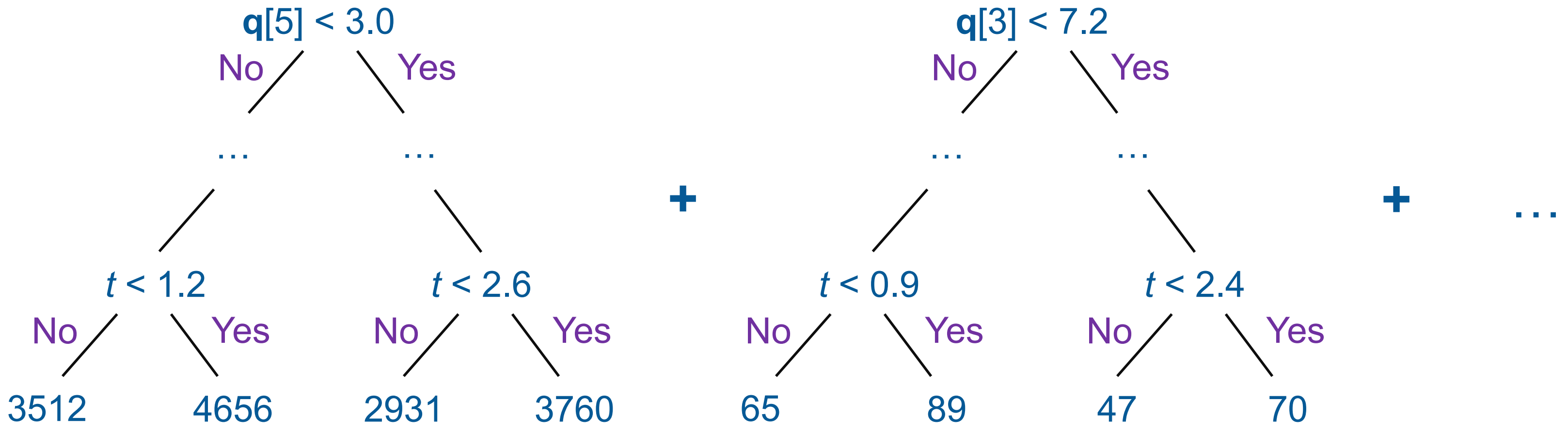
*B*(): a bandwidth function learned using query samples. The estimator is monotonic if *B*() is independent of *t*.

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

# Non-deep Regression – XGBoost



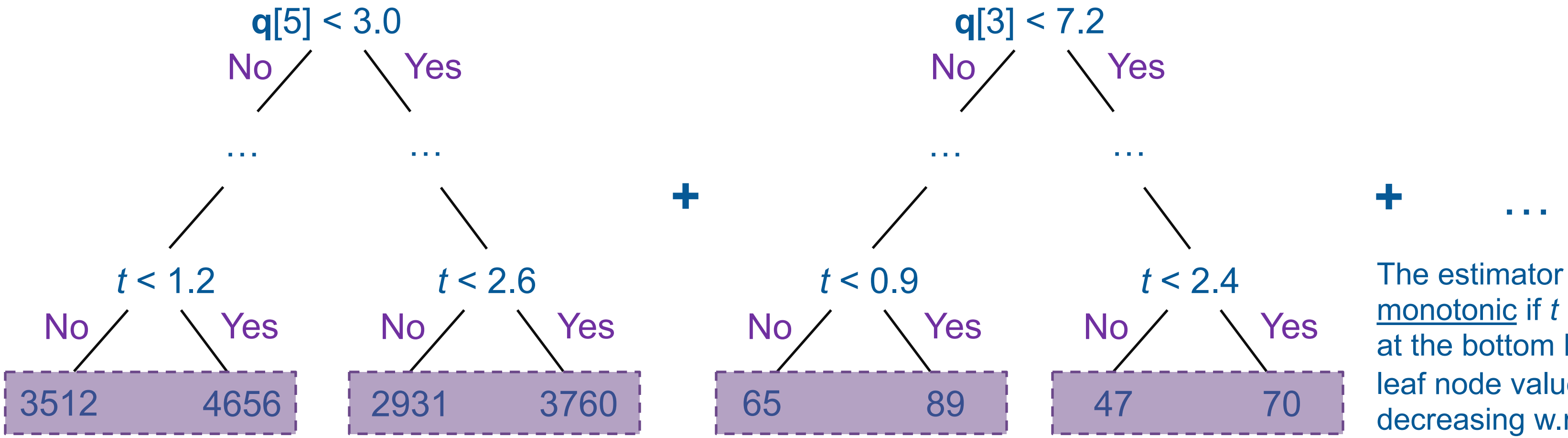
- Gradient boosting [CG16]
  - Ensemble of weak prediction models.
    - For example, decision trees, with each rule in the form of $\underline{\mathbf{q}[i] < \alpha}$ or $\underline{t < \beta}$.
  - Each model is learned to fit the residual of previous ones.



# Non-deep 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.

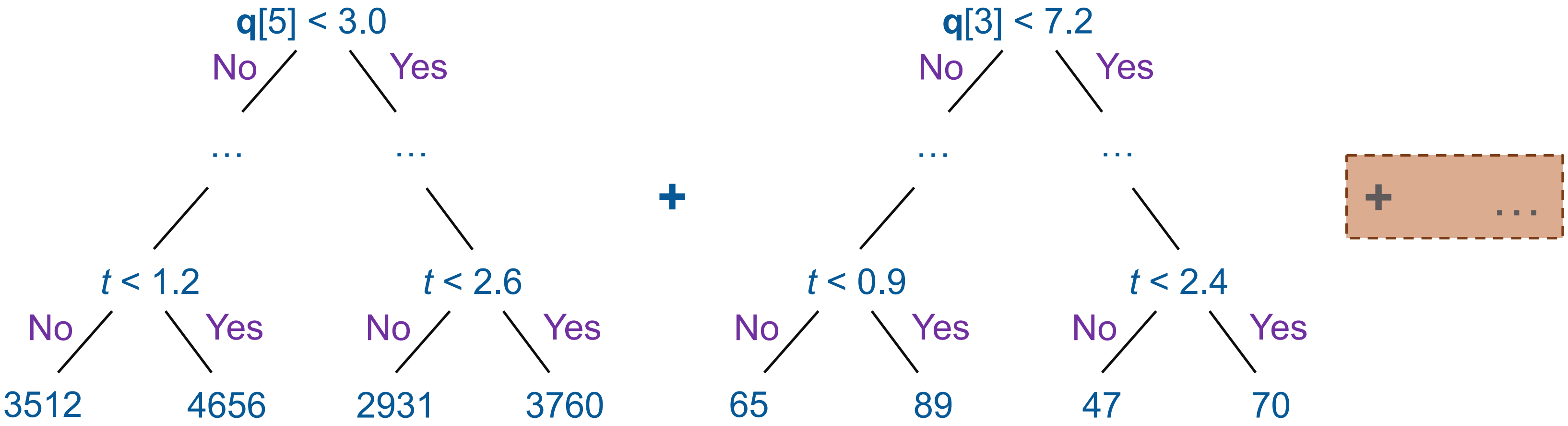



The estimator is monotonic if $t < \beta$ is only at the bottom level and leaf node values are non-decreasing w.r.t. Yes/No.

# Non-deep Regression – XGBoost



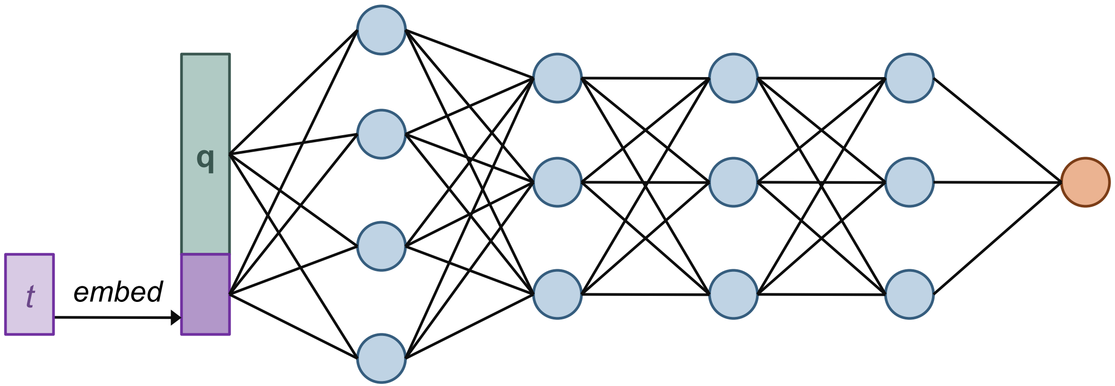
- Deal with updates.
  - It is time-consuming to retrain existing decision trees.
  - Train more decision trees to fit the residual.



# Deep Regression – Vanilla Deep Neural Network



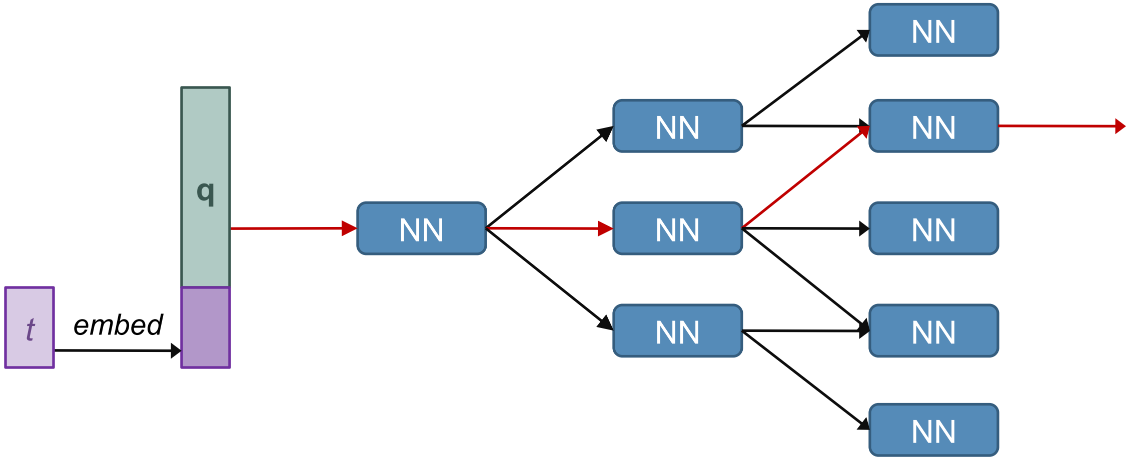
- Fully connected neural network.
  - Number of hidden layers ≈ 4.
  - For higher accuracy, embed $t$ (dim ≈ 5) and concatenate to **q** as input.
  - Non-monotonic.



# Deep Regression – Recursive Model Index



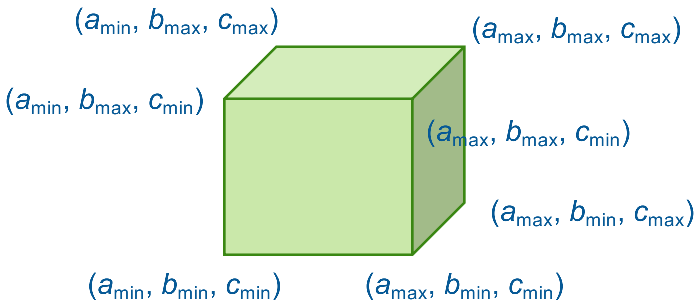
- Originally developed for range indexing in RDBMS [KBCD+18].
  - Inspired by the mixture-of-experts model.
  - Each model (e.g., a neural network) picks another one in the next stage.
  - Non-monotonic.



# Deep Regression – Deep Lattice Network



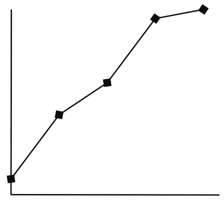
- Developed for <u>monotonic</u> regression tasks [YDCP+17].
  - Input: monotonic features + non-monotonic features.
- Components
  - Lattice: regression for a *d*-dimensional input.




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

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

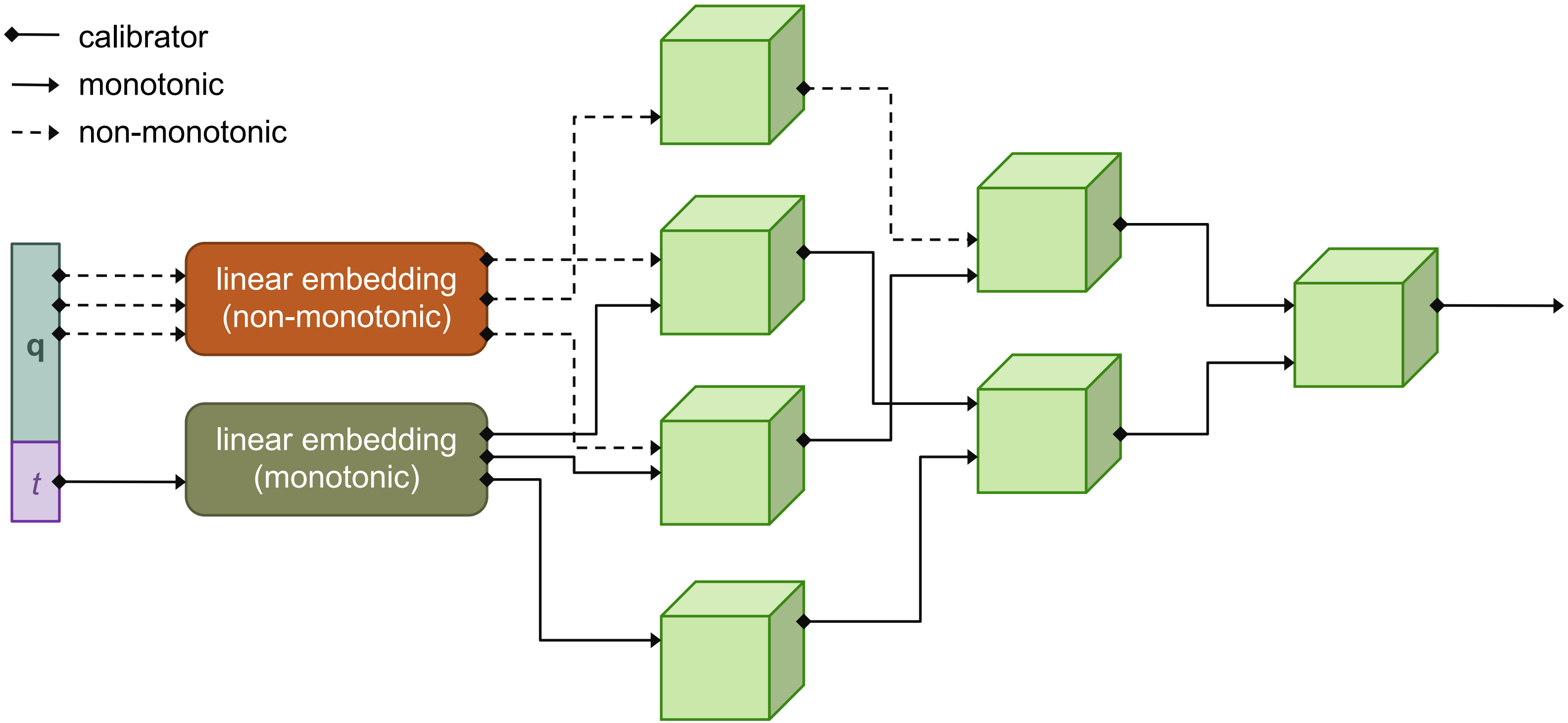
  - Calibrator: 1-dimensional non-decreasing piecewise linear function. →



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

# Deep Regression – Deep Lattice Network





# Deep Regression – Incremental Prediction (CardNet)



- Idea: use multiple regressors, each dealing with an interval [WXQC+20].
- Procedure
  - 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.

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 2.5 | -1.8 | 3.7 | -0.6 | 1.2 | → | 0 | 1 | 1 | 0 |

    - threshold $t$ → integer $\tau$
      - 0 → 0. $t_{max}$ → $\tau_{max}$. Other values are mapped in a <u>non-decreasing</u> manner → <u>monotonicity</u>.

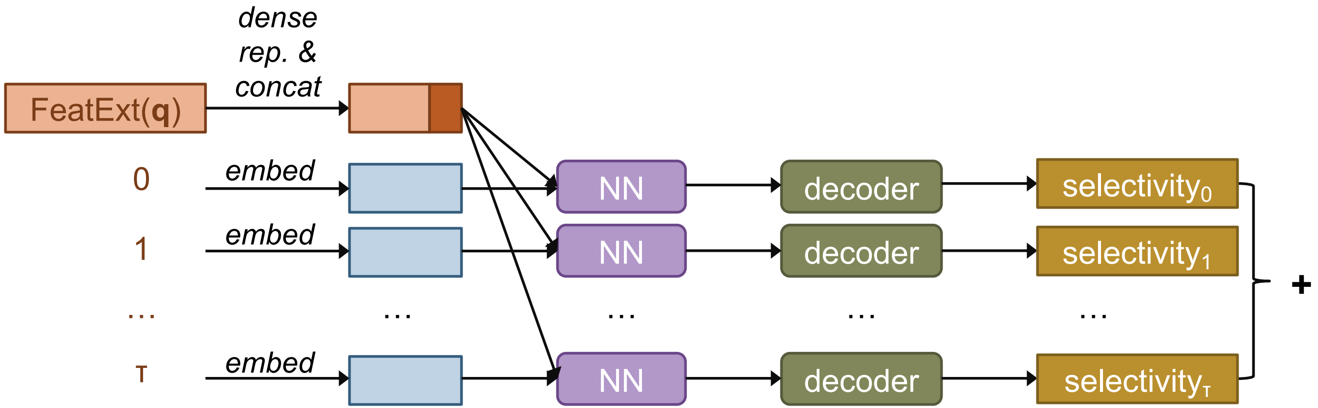
| | | |
|---|---|---|
| 2.4 | → | 6 |

  - Regression

# Deep Regression – Incremental Prediction (CardNet)



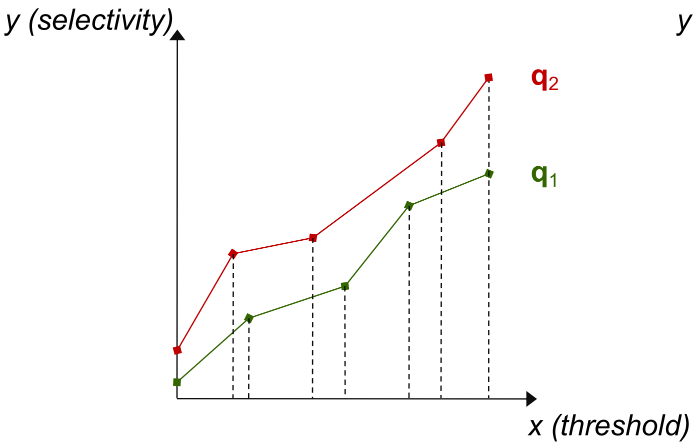
- Idea: use multiple regressors, each dealing with an interval.
- Procedure
  - Feature extraction
  - Regression
    - Use (τ + 1) regressors, each for a distance in 0, 1, … τ.



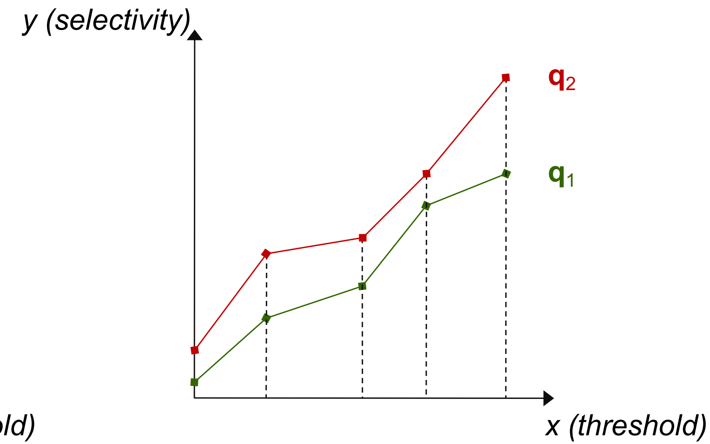
# Deep Regression – Piecewise Linear Function (SelNet)



- Query-dependent PLF v.s. query-independent PLF




query-dependent:
variable (adaptive)
control points

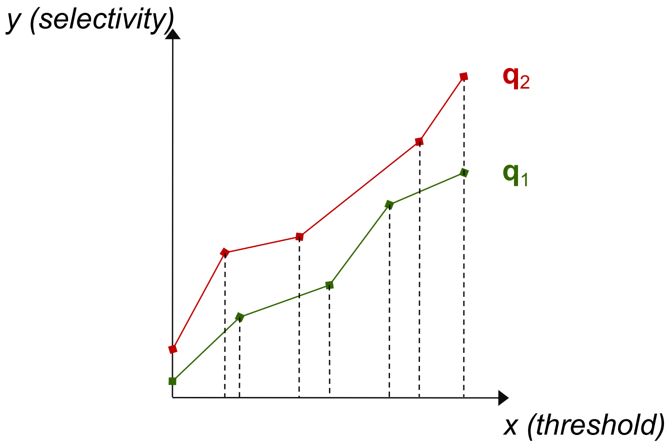


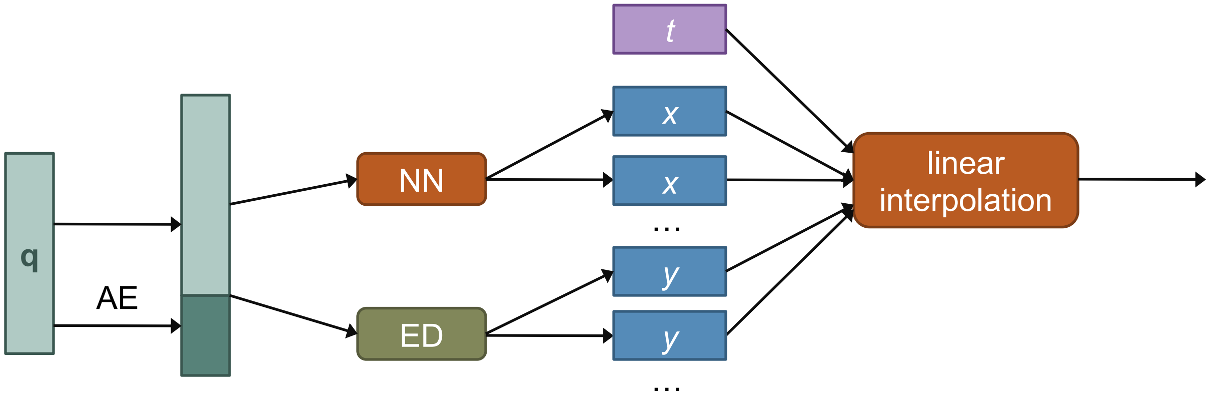

query-independent
(deep lattice network):
fixed control points

# Deep Regression – Piecewise Linear Function (SelNet)



- Learn query-dependent PLFs [WXQM+20].
  - 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*.

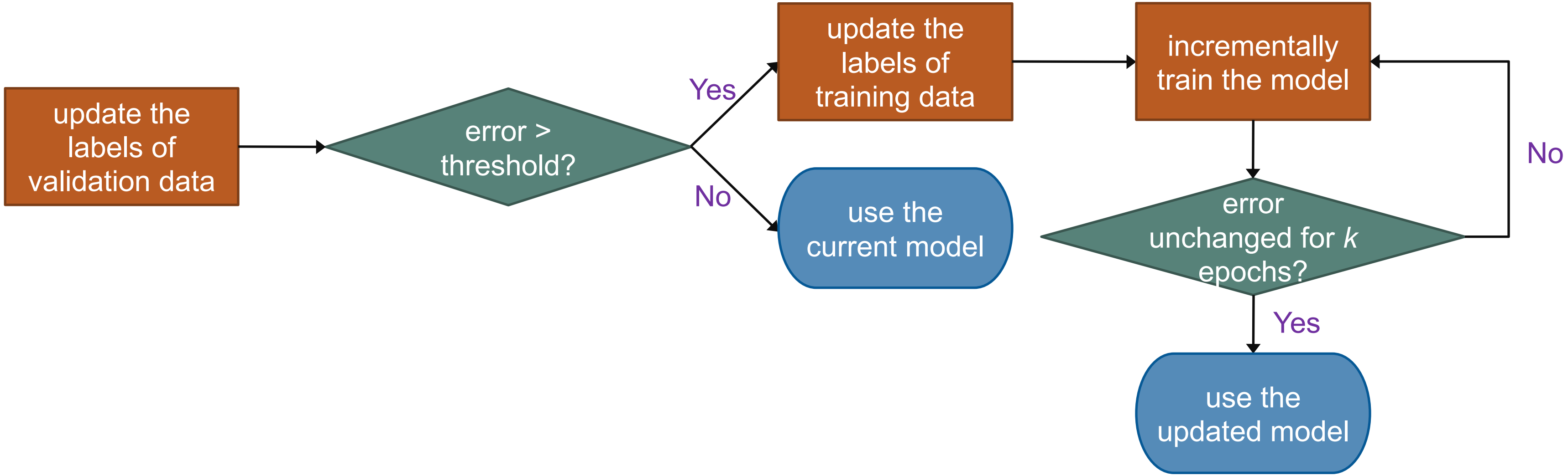


# Deep Regression – Dealing with Updates



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



# Benchmarks



- So far there are 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 |
| RMI | Medium | Medium | Slow | No | No |
| DLN | Low | Slow | Slow | No | **Yes** |
| CardNet | **High** | **Fast** | Slow | No | **Yes** |
| SelNet | **High** | Medium | Slow | No | **Yes** |

# Performance in a Query Optimizer



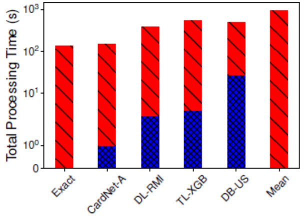
- Datasets
  - AMiner (author names & publications)
  - IMDB (cast & movie titles)
  - Attributes are pre-processed by Sentence-BERT [RG19].
- Queries
  - Conjunctive queries of 2 – 5 Euclidean distance predicates.
    - For example, *dist*(name) ≤ 0.25 AND *dist*(affiliations) ≤ 0.4 AND *dist*(research 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

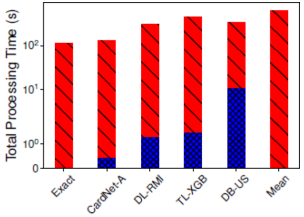


- Exact: oracle (true selectivity)
- Cardnet-A
- DL-RMI: RMI
- TL-XGB: XGBoost
- DB-US: uniform sampling
- Mean: same (mean selectivity)

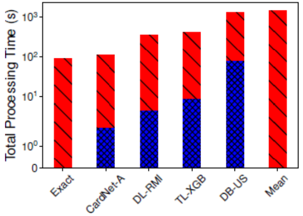
*Query processing time*:
estimation / lookup+check



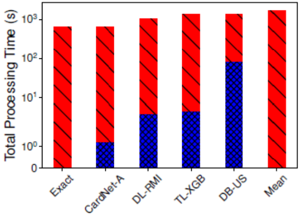

(a) Time, AMiner-Publication




(b) Time, AMiner-Author

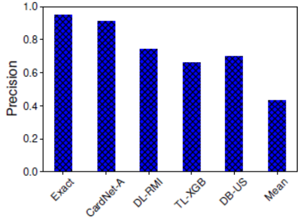



(c) Time, IMDB-Movie

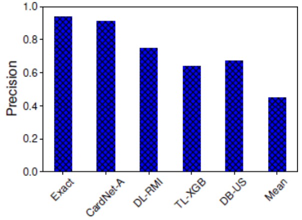



(d) Time, IMDB-Actor

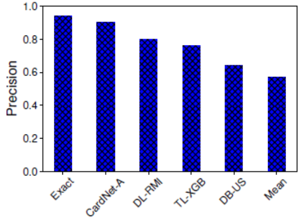
*Precision of query planning*:
% of queries on which a method picks the fastest plan



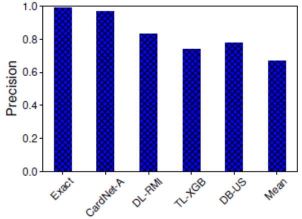

(a) Time, AMiner-Publication




(b) Time, AMiner-Author




(c) Time, IMDB-Movie




(d) Time, IMDB-Actor

# References




- 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.
- Y. Park, S. Zhong, and B. Mozafari. Quicksel: Quick selectivity learning with mixture models. In SIGMOD, pages 1017–1033, 2020.

# References




- 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.
- 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.