University of Glasgow
School of Computing Science
Essence: Pervasive & Distributed Intelligence

Data Relevance in Predictive Analytics: Thinning Big Data

YIANNIS KATHIDJIOTIS, MSC

SUPERVISOR: DR C ANAGNOSTOPOULOS

APRIL 2019@GLASGOW

Introduction

- Users find themselves executing queries that return a number of results (score) that is too low or too high compared to their task's needs.

- The execution of these "bad" queries can lead to the **waste** of network, storage, financial resources, time.

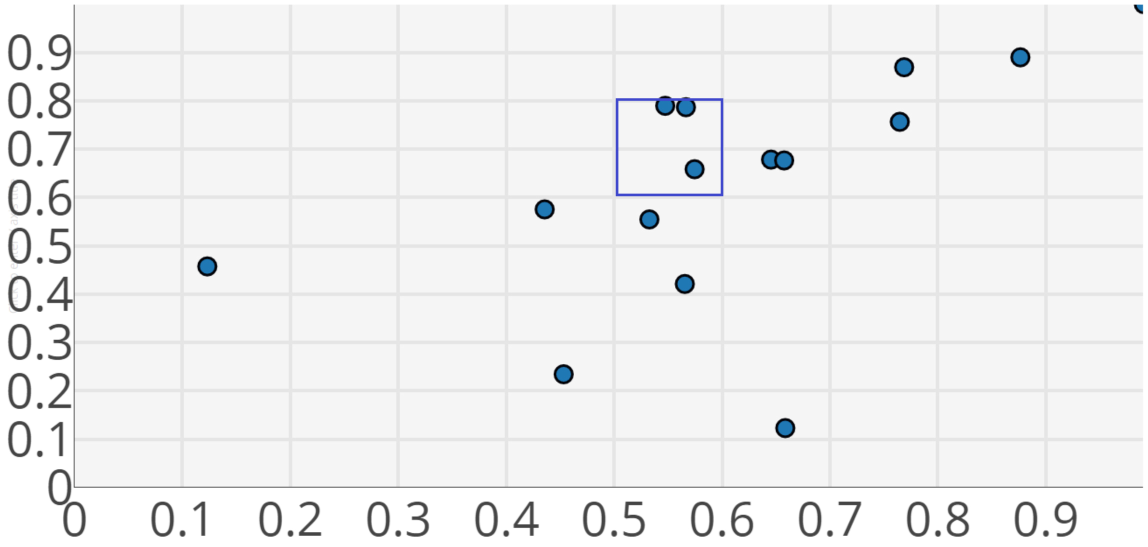
- **Hypothesis 1: Waste of resources could be avoided if we can predict the scores of queries.**

- **Hypothesis 2: Adopt score prediction to determine if a query is worth executing based on user criteria**

Queries & Dataset

- **Query: focus on range queries that are directed to a data set of d-dim real-valued data points.**

- A *d*-dim. range query is made up of *d* pairs of (min, max) values.
- Each pair corresponds to attribute *i* from the data set and declares that value *i.x* must be: $min_i \leq i.x \leq max_i$, in order for the point/tuple that contains to *i.x* to match the query.
- Whenever a tuple/point matches a query, the query score increases by 1.



Range query *q* [[0.5, 0.6], [0.6,0.8]] has a score of 3.

Query-Score Quantization

- **Step 1:** Let a set of **random** range queries of *d* [(min, max)] pairs.
- **Step 2:** Execute these queries against a normalized data set to obtain their scores.
- **Step 3:** Form the **query-score vectors**:

$$\textbf{[query, score]} = [(\boldsymbol{min_1}, \boldsymbol{max_1}), \ldots, (\boldsymbol{min_d}, \ \boldsymbol{max_d}), \textbf{score}]$$

- **Step 4:** Divide the of [query, score] vectors into a training-set (60 %) and a testing-set (40 %).
- **Step 5:** Use the training-set to quantize the vectorial space into **k-subspaces** using the **k-means algorithm**
- **Step 6:** Produce *k* vectors: [query, score] referred to as **centroids**.

Rationale: Centroid Refinement

- Locate the **two closest centroids** for each random query *q* from the testing-set.
- The two closest centroids are decided based on the **Manhattan distance** between *q* and each centroid.
- The **closest** centroid is referred to as the **winner representative** while the second closest is the **rival representative**.
- Calculate the **score error** for each representative for every *q* as the absolute difference between their scores.

- **Observation:** in many occasions the **rival** representative had the lower score error. That was the motivation for the **centroid refinement** process!
- **Centroid refinement objective is to increase the reliability of the winner representative.**
- **Hypothesis:** examine whether by increasing reliability also improves predictability.

Rationale: Centroid Refinement

- Centroid refinement involves **new** random query set (**refinement-set**) and penalty/ reward formulas.
- Penalty shift centroid's values **further** from the values of query *q*,
- **Reward** shifts centroid's values **closer** to query *q*; where *q* is a query from the refinement-set.

- **Study 1:** different approaches for how the parameter of these formulas should be acquired.
- **This parameter determines the magnitude of the penalty or reward effect.**

- **Study 2:** different variations of the centroid refinement function that make use of the two formulas to decide which is the <u>most effective</u> to be used as our final refinement approach.

Rationale: Centroid Refinement

For each query q from the refinement-set:

Option 1: If (winner representative score error < rival representative score error):
Then reward (winner representative)

Option 2: If (winner representative score error > rival representative score error):
Then reward (rival representative), penalty (winner representative)

Rationale: Score Prediction

- **Use the values of a query q’s representatives (closest centroids) for score predictions.**

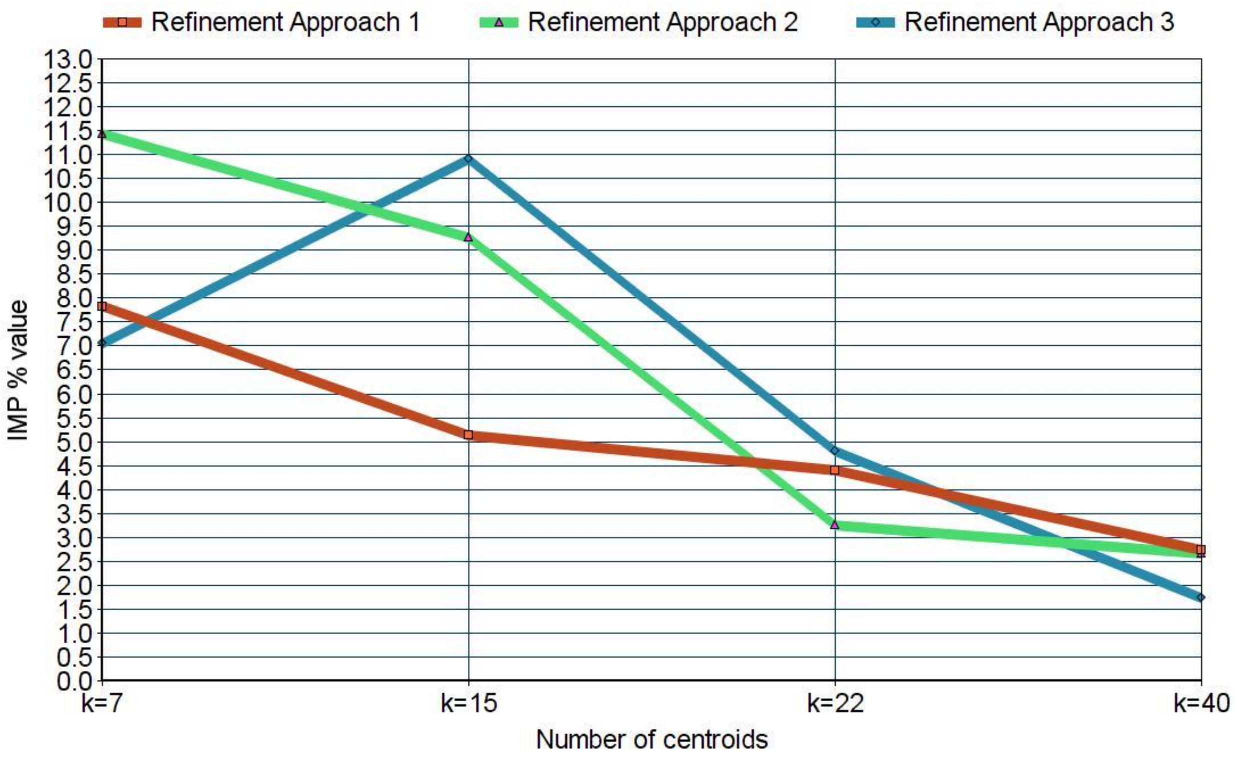
- There are three different prediction approaches that have been examined:
  1. Use the score of the winner representative as its prediction.
  2. Use a weighted sum of the scores of the winner and rival representatives as its prediction.
  3. Use a stochastic approach where the score of the winner or rival representative is used as its prediction.

Experiments

➢ Focus: on refinement, we concluded on that:

1. The parameter of the reward/ penalty formula **should depend** on the number of occupants in the cluster (that corresponds to the centroid undergoing refinement).

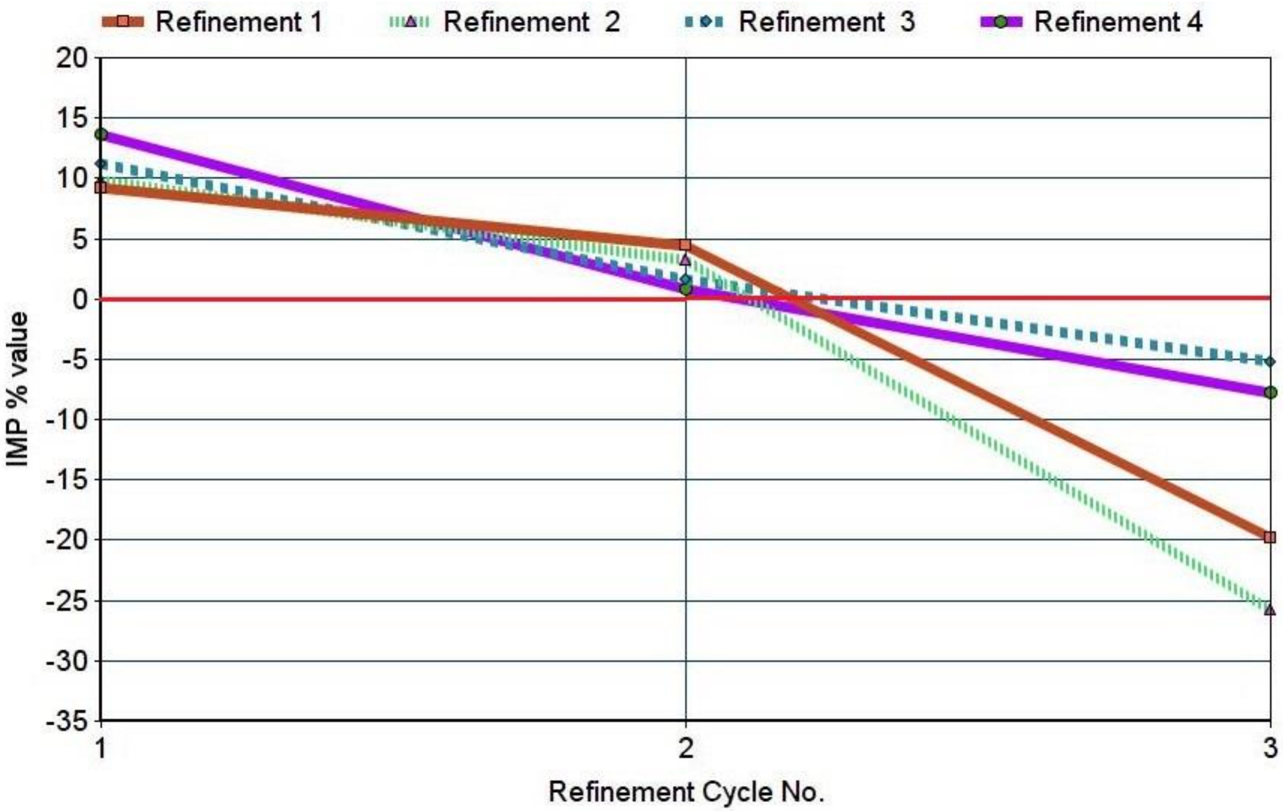
2. The **effect** of refinement decreases as *k* (number of centroids) increases.



Experiments

3. There **exists** a certain limit to how much we can increase the reliability of the winner representative;

after a certain point, the refinement can decrease its reliability!



Experiments

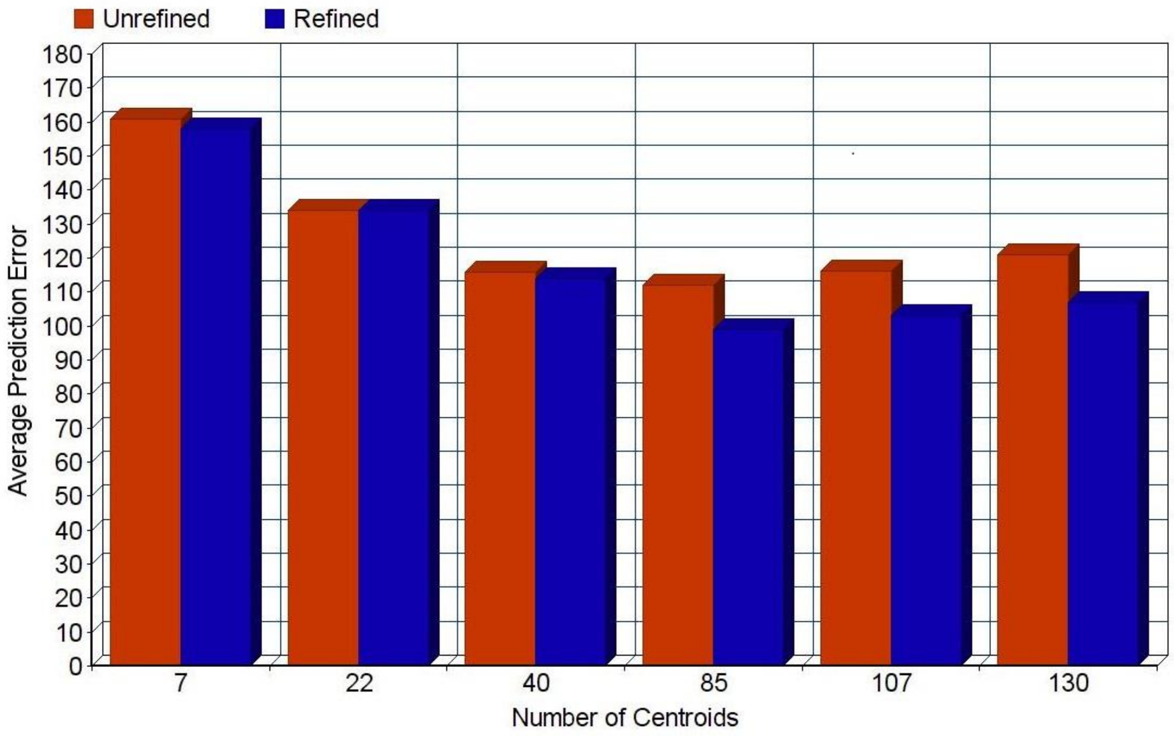
- We make separate predictions using either the **unrefined** or **refined** centroids as our prediction basis.
- This help us determine **if increased reliability improves predictability**:

Outcome 1: Using the weighted sum of the two representatives as our prediction score leads to the lowest prediction errors compared to the other approaches.

Outcome 2: Predictions can get better at higher values of *k.* Although this does not mean that predictions will get better every time *k* increases.

Outcome 3: **Increasing reliability can improve predictability!** This statement holds for the refinement-set and can be seen in the bar chart.

Outcome 4: In the case of **new** query sets the relationship between refinement and predictability is unclear as there are cases where refinement either worsens or improves predictions or in other cases its effect on predictions is too small to be deemed significant.



University of Glasgow
School of Computing Science
Essence: Pervasive & Distributed Intelligence

Experiments

- **Challenge:** "can we determine whether a query is worth executing based on score prediction and user criteria".

- Choose our most effective prediction models that make use of our most effective prediction approach at a specific *k,* using either the refined or unrefined centroids as our prediction basis.

- Measure the **sensitivity** and **specificity** for the predicted scores of a set of queries; where Sensitivity = $\frac{TP}{P}$ and Specificity = $\frac{TN}{N}$.

- **Outcome:** higher percentages were more consistent in the specificity tests. Our approach can determine **with much more confidence that a query is <u>not</u> worth executing instead of worth executing.**

- **Future work**: involve **more than** two of the closest centroids in score prediction and weight them <u>appropriately</u> to further increase predictability.

<u>Sensitivity</u>

	(0,15)	(0,50)	(0,100)	(50,100)	(200,300)	(100,300)	(100<)	(300<)
k=85 R	68.19%	83.17%	85.10%	21.89%	34.27%	68.16%	86.70%	52.67%
k=107 UR	62.80%	80.98%	86.02%	40.29%	24.73%	58.04%	84.03%	70.14%
k=107 R	72.50%	80.59%	85.60%	20.88%	28.91%	72.62%	89.26%	64.27%
k=130 UR	71.36%	86.48%	88.26%	20.82%	20.03%	63.41%	86.32%	58.48%

<u>Specificity</u>

	(0,15)	(0,50)	(0,100)	(50,100)	(200,300)	(100,300)	(100<)	(300<)
k=85 R	95.66%	91.45%	87.12%	92.89%	89.40%	79.53%	82.68%	95.84%
k=107 UR	93.04%	89.76%	85.99%	92.55%	93.23%	84.99%	86.98%	96.78%
k=107 R	97.15%	91.35%	88.28%	91.28%	93.52%	86.96%	88.01%	96.47%
k=130 UR	95.66%	91.45%	87.12%	92.84%	89.40%	79.54%	82.68%	95.84%

- Headings of rows define the prediction model. (No. of centroids, UR= Unrefined Centroids, R= Refined Centroids)
- Column headings represent user criteria. E.g. (0,15) means that: $0 \leq \text{score} \leq 15$.

Thank you!

http://www.dcs.gla.ac.uk/essence/