Database Learning: Toward a Database that Becomes Smarter Every Time

Presented by: Huanyi Chen
Where does the data come from?

- Real world
- The entire dataset follows certain underlying distribution
### Income of a shop

<table>
<thead>
<tr>
<th># of Day</th>
<th>Income (CAD)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>100</td>
</tr>
<tr>
<td>2</td>
<td>200</td>
</tr>
<tr>
<td>3</td>
<td>400</td>
</tr>
<tr>
<td>4</td>
<td>800</td>
</tr>
<tr>
<td>5</td>
<td>1600</td>
</tr>
</tbody>
</table>

#### Income of a shop per day

![Graph showing the income of a shop per day](chart.png)
## Income of a shop

<table>
<thead>
<tr>
<th># of Day</th>
<th>Income (CAD)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>100</td>
</tr>
<tr>
<td>2</td>
<td>200</td>
</tr>
<tr>
<td>3</td>
<td>400</td>
</tr>
<tr>
<td>4</td>
<td>800</td>
</tr>
<tr>
<td>5</td>
<td>1600</td>
</tr>
<tr>
<td>6</td>
<td>?</td>
</tr>
</tbody>
</table>

### Income of a shop per day

![Graph showing income of a shop per day](image-url)
Income of a shop

- $Income = 50 \times 2^n$ (n = 1, 2, 3 ...)

- No database needed if we can find the underlying distribution
Which distribution do we care?

- The exact underlying distribution that generates the entire dataset and future unseen data?
  - Not possible

- An exact underlying distribution that generates the entire dataset but excludes future unseen data?
  - Benefits nothing. One can always make a model by using every value of a column, but this model is not able to predict anything. We still need to store future data in order to answer queries.

- A possible distribution that generates the entire dataset and future unseen data!
Mismatching data

- A possible distribution that generates the entire dataset and future unseen data is not able to match every data in the dataset
  - Not work when the accurate query results needed
  - Works in Approximate query processing (AQP)
Approximate Query Processing (AQP)

- Trade accuracy for response time
- Results are based on samples
- Previous query results have no help in future queries
  - so it comes Database Learning - learning from past query answers!
Database Learning Engine: Verdict

Target

- Improve future query answers by using previous query answers from an AQP engine

Workflow

Figure 2: Workflow in Verdict. At query time, the Inference module uses the Query Synopsis and the Model to improve the query answer and error computed by the underlying AQP engine (i.e., raw answer/error) before returning them to the user. Each time a query is processed, the raw answer and error are added to the Query Synopsis. The Learning module uses this updated Query Synopsis to refine the current Model accordingly.
Verdict

- A query is decomposed into possibly multiple query snippets
  - the answer of a snippet is a single scalar value

Figure 3: Example of a query’s decomposition into multiple snippets.
Verdict

- A query is decomposed into possibly multiple query snippets
- Verdict exploits potential correlations between snippet answers to infer the answer of a new snippet

### # of Day | Income (CAD)
---|---
1 | 100
2 | 200
3 | 400
4 | 800
5 | 1600

old avg and new avg are correlated
Inference

- observations + rules = prediction

<table>
<thead>
<tr>
<th>Observations</th>
<th>Rules</th>
<th>Prediction</th>
</tr>
</thead>
<tbody>
<tr>
<td>Shop Income</td>
<td>100, 200, 400, 800, 1600</td>
<td>$Income = 50 \times 2^n$</td>
</tr>
<tr>
<td>Fibonacci</td>
<td>Initial: 1, 1</td>
<td>$F_n = F_{n-2} + F_{n-1}$</td>
</tr>
<tr>
<td></td>
<td>Past snippet answers from AQP + AQP answer for the new snippet</td>
<td>Maximize the conditional joint probability distribution function (pdf)</td>
</tr>
</tbody>
</table>
If we have $f(\theta_1 = \theta'_1, \ldots, \theta_{n+1} = \theta'_{n+1}, \bar{\theta}_{n+1} = \bar{\theta}'_{n+1})$

then the prediction is the value of $\bar{\theta}'_{n+1}$ that maximizes

$$f(\bar{\theta}_{n+1} = \bar{\theta}'_{n+1} | \theta_1 = \theta_1, \ldots, \theta_{n+1} = \theta_{n+1})$$
Inference: pdf

- How to find the pdf?
  - maximum entropy (ME) principal
    - \[ h(f) = - \int f(\tilde{\theta}) \cdot \log f(\tilde{\theta}) d\tilde{\theta}, \quad \text{where } \tilde{\theta} = (\theta_1', ..., \theta_{n+1}', \bar{\theta}_{n+1}') \]

- The joint pdf maximizing the above entropy differs depending on the kinds of given testable information
  - Verdict uses the first and the second order statistics of the random variables: mean, variances, and covariances.
Lemma 1. Let \( \mathbf{\theta} = (\theta_1, \ldots, \theta_{n+1}, \bar{\theta}_{n+1})^\top \) be a vector of \( n+2 \) random variables with mean values \( \bar{\mu} = (\mu_1, \ldots, \mu_{n+1}, \bar{\mu}_{n+1})^\top \) and a \((n+2)\times(n+2)\) covariance matrix \( \Sigma \) specifying their variances and pairwise covariances. The joint pdf \( f \) over these random variables that maximizes \( h(f) \) while satisfying the provided means, variances, and covariances is the following function:

\[
f(\bar{\mathbf{\theta}}) = \frac{1}{\sqrt{(2\pi)^{n+2} |\Sigma|}} \exp\left(-\frac{1}{2} (\bar{\mathbf{\theta}} - \bar{\mathbf{\mu}})^\top \Sigma^{-1} (\bar{\mathbf{\theta}} - \bar{\mathbf{\mu}})\right)
\]

and this solution is unique.
Inference: model-based answer and error

Generally

\[ \bar{\theta}_{n+1} = \text{Arg Max}_{\bar{\theta}'_{n+1}} f(\bar{\theta}'_{n+1} | \theta_1 = \theta_1, \ldots, \theta_{n+1} = \theta_{n+1}) \]

Computing above conditional pdf may be a computationally expensive task
Inference: model-based answer and error

However, computing the conditional pdf in lemma 1 is not expensive and computable; the result is another normal distribution. The mean $\mu_c$ and variance $\sigma_c^2$ are given by:

\[
\begin{align*}
\mu_c &= \bar{\mu}_{n+1} + \bar{k}_{n+1}^\top \Sigma_{n+1}^{-1}(\tilde{\theta}_{n+1} - \bar{\mu}_{n+1}) \\
\sigma_c^2 &= \bar{k}^2 - \bar{k}_{n+1}^\top \Sigma_{n+1}^{-1} \bar{k}_{n+1}
\end{align*}
\]

where:

- $\bar{k}_{n+1}$ is a column vector of length $n + 1$ whose $i$-th element is $(i, n + 2)$-th entry of $\Sigma$;
- $\Sigma_{n+1}$ is a $(n + 1) \times (n + 1)$ submatrix of $\Sigma$ consisting of $\Sigma$’s first $n + 1$ rows and columns;
- $\tilde{\theta}_{n+1} = (\theta_1, \ldots, \theta_{n+1})^\top$;
- $\bar{\mu}_{n+1} = (\mu_1, \ldots, \mu_{n+1})^\top$; and
- $\bar{k}^2$ is the $(n+2, n+2)$-th entry of $\Sigma$. 


Inference: model-based answer and error

- Model-based answer
  - $\dot{\theta}_{n+1} = \mu_c$

- Model-based error
  - $\dot{\beta}_{n+1} = \sigma_c$

- Improved answer and error
  - $(\hat{\theta}_{n+1}, \hat{\beta}_{n+1}) = (\ddot{\theta}_{n+1}, \ddot{\beta}_{n+1})$ (if validation succeed)
  - $(\hat{\theta}_{n+1}, \hat{\beta}_{n+1}) = (\theta_{n+1}, \beta_{n+1})$ (if validation failed, return AQP answers)
Inference: means, variances, and covariances

- mean ($\mu$)
  - the arithmetic mean of the past query answers for the mean of each random variable, $\theta_1, \ldots, \theta_{n+1}, \bar{\theta}_{n+1}$.

- variances, and covariances ($\Sigma$)
  - the covariance between two query snippet answers is computable using the covariances between the attribute values involved in computing those answers.
Inference: means, variances, and covariances

<table>
<thead>
<tr>
<th># of Day</th>
<th>Income (CAD)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>100</td>
</tr>
<tr>
<td>2</td>
<td>200</td>
</tr>
<tr>
<td>3</td>
<td>400</td>
</tr>
<tr>
<td>4</td>
<td>800</td>
</tr>
<tr>
<td>5</td>
<td>1600</td>
</tr>
</tbody>
</table>

old avg and new avg are correlated
Inference: means, variances, and covariances

Inter-tuple Covariances

<table>
<thead>
<tr>
<th># of Day</th>
<th>Income (CAD)</th>
<th>Income (CAD)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>2</td>
<td>200</td>
<td>200</td>
</tr>
<tr>
<td>3</td>
<td>400</td>
<td>400</td>
</tr>
<tr>
<td>4</td>
<td>800</td>
<td>800</td>
</tr>
<tr>
<td>5</td>
<td>1600</td>
<td>1600</td>
</tr>
</tbody>
</table>
Inference: means, variances, and covariances

- Estimate the inter-tuple covariances
  - analytical covariance functions
    - squared exponential covariance functions: capable of approximating any continuous target function arbitrarily closely as the number of observations (here, query answers) increases
  - compute variances, and covariances \((\Sigma)\) efficiently
Experiments

- Up to $23 \times$ speedup for the same accuracy level
- Small memory and computational overhead
Summary

- An idea: Database Learning
  - learning from past query answers

- An implementation: Verdict
  - Given mean, variances, and covariances
  - Apply maximum entropy principal
  - Find a joint probability distribution function
  - Improve answer and error based on conditioning on snippet answers
  - https://verdictdb.org
Q & A

- Using testable information other than or in addition to mean, variances, covariances?
- Are there any other possible inferential techniques?
- Can we cut out training phase?