УДК [611.018.51+615.47]:612.086.2

MACHINE LEARNING AND LAMBDA ARCHITECTURE METHODS APPLICATION TO OPTIMIZE POSTGRESQL DATABASE PERFORMANCE

Y. Sharayeu
Master student of Electronic Computing Machines Department of BSUIR

Belarusian State University of Informatics and Radioelectronics, Republic of Belarus
E-mail: eugen.sharayev@gmail.com

Y. Sharayeu
Master student of Electronic Computing Machines Department of Belarusian State University of Informatics and Radioelectronics. Specialty: «Technologies of virtualization and cloud computing». Graduated from the first stage of higher education of BSUIR by specialty "Computing machines, systems and networks".

Abstract. Here is described the application of methods of optimizing POSTGRESQL database using machine learning algorithms and lambda architecture principles.

Keywords: machine learning, relational databases, lambda architecture.

Introduction. As far as SQL is a declarative language, the performance of queries has to be ensured on the database level. There is a module called Planner/Optimizer that is designed to improve queries performance by finding the best execution plan for a given query. Execution plans are defined as a set of operations that the database performs to access and process stored data. In order to estimate which plan to pick Planner/Optimizer module use a value called plan cost. This value is a sum of each operation that is going to be performed during query execution. Each operation cost is evaluated depending on a cardinality value (a number of tuples which are going to be processed during a plan step) and operation cost constants. Operation cost constants are required to estimate the cost of a query depending on a kind of computation (*seq_page_cost* refers to a cost of sequential reading of a page from a disk and is equal to 1.0, *cpu_tuple_cost* defines a cost of processing a table row on a CPU and is equal to 0.01 etc.). The approximate value of cardinality is fetched from statistics that the database collects, stores and constantly updates when data changes.

Functional dependencies. By default, PostgreSQL assumes that columns don't have functional dependencies between themselves. Thus some performance issues may occur in cases like querying *users* table with filters on *age* and *married* attributes. In case of the uniform distribution of users by age and assumption that you can't get married until you turn 18, database will expect queries *SELECT * FROM users WHERE age BETWEEN 0 AND 17 AND married IS TRUE* and *SELECT * FROM users WHERE age BETWEEN 18 AND 36 AND married IS TRUE* to have equal cardinality estimates, but in fact the first query won't return any records, while the second will return a few results. This will lead to performance degradation of the first query. It may be not so

problematic for small databases and straightforward queries, but in the case of big data and sophisticated selects, it may result in huge delays and even timeout errors.

Multivariate statistics. In order to mitigate functional dependencies problem, PostgreSQL has a feature called multivariate statistics [1]. This kind of statistics is a multidimensional array (as opposed to regular one-dimensional statistics) that is used when queries with filters on attributes with functional dependencies are executed.

Applying lambda architecture. As an alternative to multivariate statistics, machine learning model can be used to predict cardinality values. There are three approaches to do this: to leverage cardinality results after each query to train model, to use a separate worker process to constantly query the database and receive actual cardinality results for training and to use a mixed approach. The first approach has the following advantages: it requires only enabling the corresponding plugin and it doesn't affect the database performance as it doesn't perform any extra queries that could utilize CPU, interfere database and filesystem level caches etc. The first drawback is that only specific models like knn (k-nearest neighbors) or decision trees can be used as far as they are able to be trained fast (and even simultaneously in case of knn) during a single epoch. Another drawback is that chosen models have to be able to address concept drift that refers to prediction errors due to data changes (as an example, knn can mitigate them by storing only last N neighbors). The second approach gives an ability to choose any model including high-precision deep learning models. Issues with concept drift can be mitigated by constantly re-fitting the data. Impact on CPU and cache can be eliminated by creating a replica on another server only for training the model and then propagate it to other database servers. Drawbacks of this method are increasing administration complexity, having an extra server and possible obsolescence of model predictions due to concept drift. The third option leverages the advantages of both approaches by using an architecture similar to lambda architecture for big data processing [3]. In terms of lambda architecture, speed layer is implemented using the first approach with incrementally learning models and batch layer is implemented using more precise models that are constantly re-trained on the replica. The decision on what model to use for the prediction on a current query (from speed or batch layer) can be done similar to the implementation of statistics expiration in Oracle Database [4]: consider batch layer model view stale, if more than 10% of rows are changed since the last re-fit on the replica.

References

[1]. PostgreSQL 10.5 Documentation. – P. 3370.

[2]. A survey on concept drift adaptation / J. Gama [et al.] // ACM Computing Surveys. – 2014. – Vol. 46, № 4. – P. 1-37.

[3]. Marz, N. Big Data: Principles and Best Practices of Scalable Real-time Data Systems / N. Marz, J. Warren. – Manning, 2015. – 308 p.

[4]. Oracle SQL Recipes: A Problem-Solution Approach / G. Allen [et al.]. – Apress, 2010. – 550 p.

ПРИМЕНЕНИЕ МЕТОДОВ МАШИННОГО ОБУЧЕНИЯ И ЛЯМБДА-АРХИТЕКТУРЫ ДЛЯ ОПТИМИЗАЦИИ БАЗЫ ДАННЫХ POSTGRESQL

Е.В. Шараев
Магистрант кафедры ЭВМ БГУИР

Белорусский государственный университет информатики и радиоэлектроники,
Республика Беларусь
E-mail: eugen.sharayev@gmail.com

Аннотация. Описано применение методов оптимизации базы данных PostgreSQL посредством алгоритмов машинного обучения с использованием принципов лямбда-архитектуры.

Ключевые слова: машинное обучение, реляционные базы данных, лямбда архитектура.