UNIVERSIDADE ESTADUAL DO CEARÁ

CENTRO DE CIÊNCIAS E TECNOLOGIA

PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

MESTRADO ACADÊMICO EM CIÊNCIA DA COMPUTAÇÃO

LÁZARO JOÃO CÁ

UMA ABORDAGEM PARA MODELAGEM DE DESEMPENHO PARA SISTEMAS DE BANCO DE DADOS NEWSQL E COMPARAÇÃO DE MODELOS GERADOS COM DIFERENTES TÉCNICAS DE APRENDIZADO DE MÁQUINA

FORTALEZA – CEARÁ

2018

LÁZARO JOÃO CÁ

UMA ABORDAGEM PARA MODELAGEM DE DESEMPENHO PARA SISTEMAS DE BANCO DE DADOS NEWSQL E COMPARAÇÃO DE MODELOS GERADOS COM DIFERENTES TÉCNICAS DE APRENDIZADO DE MÁQUINA

Dissertação apresentada ao Curso de Mestrado Acadêmico em Ciência da Computação do Programa de Pós-Graduação em Ciência da Computação do Centro de Ciências e Tecnologia da Universidade Estadual do Ceará, como requisito parcial à obtenção do título de mestre em Ciência da Computação. Área de Concentração: Redes e Sistemas Distribuídos

Orientador: Prof. Dr. Marcial Porto Fernandez

FORTALEZA – CEARÁ

2018

LÁZARO JOÃO CÁ

UMA ABORDAGEM PARA MODELAGEM DE DESEMPENHO PARA SISTEMAS DE BANCO DE DADOS NEWSQL E COMPARAÇÃO DE MODELOS GERADOS COM DIFERENTES TÉCNICAS DE APRENDIZADO DE MÁQUINA

Dissertação apresentada ao Curso de Mestrado Acadêmico em Ciência da Computação do Programa de Pós-Graduação em Ciência da Computação do Centro de Ciências e Tecnologia da Universidade Estadual do Ceará, como requisito parcial à obtenção do título de mestre em Ciência da Computação. Área de Concentração: Redes e Sistemas Distribuídos

Aprovada em:

BANCA EXAMINADORA

Prof. Dr. Marcial Porto Fernandez (Orientador)
Universidade Estadual do Ceará – UECE

Prof. Dr. Paulo Henrique Mendes Maia
Universidade Estadual do Ceará – UECE

Prof. Dr. Leonardo Oliveira Moreira
Universidade Federal do Ceará – UFC

Ao meu falecido pai, por investir em mim. Sua capacidade de acreditar em mim e seu conselho foi que deram, em alguns momentos, a esperança para seguir. Se eu cheguei onde estou hoje, é porque sua presença significou segurança e certeza de que não estou sozinho nessa caminhada.

AGRADECIMENTOS

Primeiramente a Deus que permitiu que tudo isso acontecesse, ao longo de minha vida, e não somente nestes anos como universitária, mas que em todos os momentos é o maior mestre que alguém pode conhecer.

Aos meus pais, pelo amor, incentivo e apoio incondicional.

Obrigado meus irmãos, que nos momentos de minha ausência dedicados ao estudo superior, sempre fizeram entender que o futuro é feito a partir da constante dedicação no presente!

A esta universidade, seu corpo docente, direção e administração que oportunizaram a janela que hoje vislumbro um horizonte superior, eivado pela acendrada confiança no mérito e ética aqui presentes.

Agradeço a todos os professores por me proporcionar o conhecimento não apenas racional, mas a manifestação do caráter e afetividade da educação no processo de formação profissional, por tanto que se dedicaram a mim, não somente por terem me ensinado, mas por terem me feito aprender. a palavra mestre, nunca fará justiça aos professores dedicados aos quais sem nominar terão os meus eternos agradecimentos.

"É melhor lançar-se à luta em busca do triunfo mesmo expondo-se ao insucesso, que formar fila com os pobres de espírito, que nem gozam muito nem sofrem muito; E vivem nessa penumbra cinzenta sem conhecer nem vitoria nem derrota."

(Franklin Roosevelt)

RESUMO

Com o crescimento do uso de aplicativos móveis, redes sociais e muitas outras aplicações web, surgiu uma preocupação com a enorme variedade de dados usados por essas aplicações, com complexidade de esquemas de representação desses dados da qual Sistemas de Gerenciamento de Banco de Dados (SGBDs) Relacionais tradicionais (i.e., SGBDs SQL) não conseguem gerenciar de forma eficiente. Um dos principais avanços na resolução desses problemas foi o surgimento de tecnologias de banco de dados alternativas como o NoSQL e, mais recentemente, o NewSQL, projetados para lidar com o volume, a variedade, a velocidade e a variabilidade das coleções dados. O NewSQL é uma classe de modernos SGBD Relacionais que fornecem o mesmo desempenho escalável dos sistemas NoSQL para cargas de trabalho de leitura e gravação ao Processamento de Transações On-Line (OLTP), enquanto ainda mantêm as garantias Atomicidade, Consistência, Isolamento e Durabilidade (ACID) de um SGBD tradicional. Este trabalho propõe uma abordagem de modelagem do desempenho baseada em técnicas de aprendizado de máquina para sistemas de bancos de dados NewSQL. Essa abordagem leva em consideração cargas de trabalho heterogêneas e é capaz de capturar os efeitos não-lineares dos aspectos de concorrência e de distribuição sobre o desempenho. Ela é avaliada num ambiente real, usando um servidor de 64 GB DDR4 RDIMM, 256 GB SSD, 2 TB SAS, 2 Processador Intel Xeon Silver 4116 e 4 Gbps. Também foi feita comparação do desempenho entre modelos gerados com diferentes técnicas de aprendizado de máquina.

Palavras-chave: Modelagem de Desempenho. Aprendizado de Máquina. NewSQL. SLA

ABSTRACT

According to increasing use of mobile applications, social networks and many other web applications, we have a new bother about the enormous variety of this these applications data, increasing complexity of it representation bt relational DBMS that can't manage efficiently. One of the great advances to solve these problems is the emergence of alternative database technologies such as NoSQL and more recently we have the NewSQL designed to handle massive volume, variety, high speed, and great variability of data collections. NewSQL is a kind of the moderns Relational DBMSs that provide the same scalable performance of NoSQL systems for read and write workloads to Online Transaction Processing (OLTP) while it still maintaining the ACID (Atomicity, Consistency, Isolation, and Durability) waranties of a traditional DBMS. This thesis proposes an approach to performance modeling based on machine learning techniques for NewSQL database systems. This approach uses heterogeneous workload to capture non-linear effects of competition and distribution of performance aspects. The tests was performed in a real environment using a server from the Department of Academic Master in Computer Science of the State University of Ceará. Performance comparison was performed between models obtained from different machine learning techniques.

Keywords: Performance Modeling. Machine Learning. NewSQL. SLA

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE QUADROS

LISTA DE ABREVIATURAS E SIGLAS

ACID	Atomicidade, Consistência, Isolamento e Durabilidade
AM	Aprendizado de Máquina
API	Interface de Programação de Aplicação
DBA	Administrador de Banco de Dados
OLTP	Processamento de Transações On-Line
RI	Restrições de Integridade
SGBD	Sistema de Gerenciamento de Banco de Dados
SLA	Acordo de Nível de Serviço
SQL	Structured Query Language
TI	Tecnologia de Informação

LISTA DE SÍMBOLOS

R^2	Coeficiente de determinação do erro
Γ	Coeficiente de reflexão

SUMÁRIO

1 INTRODUÇÃO

1.1 MOTIVAÇÃO

Os sistesmas de gerenciamento de dados convencionais, por exemplo, Sistemas de Gerenciamento de Banco de Dados (SGBDs) Relacional clássico ou mecanismos de busca convencionais não conseguem gerenciar de forma eficiente as enormes variedades de dados usados por aplicações atualmente e seus complexos esquemas de representação (MONIRUZZAMAN, 2014). Um dos principais avanços na resolução desse problema foi o surgimento de uma infinidade de SGBDs distribuídos que apareceram no cenário dos SGBDs como alternativa em relação aos sistemas relacionais tradicionais, que podem ser classificado em NoSQL e NewSQL. Estas são uma ampla classes de SGBDs que diferem significativamente do modelo clássico de SGBDs Relacionais.

Este trabalho aborda singularmente sistemas NewSQL, que é uma das alternativas em relação aos sistemas relacionais tradicionais, que lida com Processamento de Transações On-Line (OLTP). Segundo Pavlo e Aslett (2016) esses sistemas permitem que os aplicativos executem um grande número de transações simultâneas para manipular novas informações e modificar o estado do banco de dados usando o SQL, invés de uma Interface de Programação de Aplicação (API) proprietária. Se um aplicativo usa um SGBD NewSQL, os desenvolvedores não precisam desenvolver uma lógica para lidar com as atualizações eventualmente consistentes, como fazem em sistemas NoSQL. Eles são baseados em arquiteturas distribuídas que operam em recursos não compartilhados e contêm componentes para suportar o controle de concorrência de vários nós, tolerância a falhas, controle de fluxo e processamento de consulta distribuída.

Usuários de serviços esperam que os provedores assegurem a qualidade de serviço, nomeadamente nos aspectos de desempenho e de disponibilidade inerentes às aplicações (FARIAS *et al.*, 2017). É frequentemente verificado que os provedores geralmente garantam a disponibilidade, mas nem sempre garantem o desempenho como deveria. Portanto, é categórico que estes ofereçam a qualidade baseado nos aspectos de desempenho, com objetivo de melhorar a experiência do usuário final das aplicações. Desse modo, foi proposta uma abordagem de modelagem para desempenho baseada em técnicas de aprendizado de máquina para sistemas de bancos de dados NewSQL. Essa abordagem leva em consideração cargas de trabalho heterogêneas e é capaz de capturar os efeitos não-lineares dos aspectos de concorrência e distribuição sobre o desempenho.

Esta abordagem é baseada no trabalho de Farias *et al.* (2017) que propôs uma modelagem de desempenho para sistemas de banco de dados NoSQL, usando técnicas de aprendizado de máquina, na qual a metodologia foi aproveitada neste trabalho e, obviamente, difere em todos os aspectos característicos de sistema de banco de dados NewSQL. O *benchmark* usado neste trabalho foi o mesmo utilizado por Farias *et al.* (2017), mas configurando-o com *drivers* específicos para sistemas de banco de dados NewSQL. Também foram propostas outras técnicas de aprendizado de máquina diferentes das utilizadas no trabalho de Farias *et al.* (2017), das quais foram gerados modelos de desempenho. Foram implementadas, também neste trabalho, as técnicas utilizadas por Farias *et al.* (2017), gerando os modelos de desempenho também a partir destas técnicas. Depois fez-se a comparação do desempenho entre os modelos gerados usando as técnicas propostas neste trabalho com os modelos gerados pelas técnicas do trabalho de Farias *et al.* (2017).

Os algoritmos de aprendizado de máquina permitem criar modelo preditivos usando dados de exemplos ou experiências passadas (ALPAYDIN, 2014), da qual podem ser utilizados para modelar o desempenho, visto que aspectos de concorrência e de distribuição podem degradar o desempenho de modo não-linear, portanto desempenho de um SGBD NewSQL pode não ser linearmente correlacionado com a quantidade de recursos alocados, o que configura ainda mais desafio para abordagens de modelagem do desempenho (FARIAS *et al.*, 2017), que tenha a habilidade de estimar os impactos de execução concorrentes de requisições em um carga de trabalho em evolução ao longo do tempo.

1.2 OBJETIVOS

1.2.1 Objetivo Geral

O objetivo geral deste trabalho é propor uma solução para modelagem de desempenho que seja capaz de estimar o comportamento de cargas de trabalho com certo tamanho do servidor (i.e., quantidade de servidores disponíveis) para sistemas de bancos de dados NewSQL e encontrar a técnica de aprendizado de máquina mais adequada para fazer predição do comportamento de cargas de trabalho.

1.2.2 Objetivos Específicos

Os seguintes objetivos específicos foram estabelecidos para alcançar o objetivo geral deste trabalho:

a) Propor uma modelagem de desempenho baseada no trabalho de Farias *et al.* (2017);
b) Investigar a viabilidade de utilizar técnicas de aprendizado de máquina para modelagem de desempenho para sistemas de banco de dados NewSQL;
c) Avaliar e escolher a técnica de aprendizado de máquina mais adequada para predição do comportamento de cargas de trabalho; e
d) Implementar um protótipo de sistema para avaliar o desempenho de sistemas de banco de dados NewSQL usando *benchmark*.

2 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo é realizado um estudo detalhado sobre cada conteúdo abordado neste trabalho. É apresentado conceito de banco de dados, especificamente seus tipos, inclusive NewSQL, onde é feita um estudo de forma resumida sobre NuoDB, um dos banco de dados NewSQL popular. Em seguida, é abordado o conceito de Aprendizado de Máquina AM.

2.1 BANCO DE DADOS

Os bancos de dados e os sistemas de bancos de dados se tornaram componentes essenciais no cotidiano da sociedade moderna. No decorrer do dia, a maioria de nós se depara com atividades que envolvem alguma interação com os bancos de dados. Por exemplo, se formos ao banco para efetuarmos um depósito ou retirar dinheiro, se fizermos reservas em um hotel ou para a compra de passagens aéreas, se acessarmos o catálogo de uma biblioteca informatizada para consultar uma bibliografia, ou se comprarmos produtos como livros, brinquedos ou computadores de um fornecedor por intermédio de sua página *Web*, muito provavelmente essas atividades envolverão uma pessoa ou um programa de computador que acessará um banco de dados (ELMASRI; NAVATHE; MORAIS, 2005).

Um SGBD é uma coleção de dados inter-relacionados e um conjunto de programas para acessar esses dados. A coleta de dados, geralmente referida como o banco de dados, contém informações relevantes para uma organização. O objetivo principal de um SGBD é fornecer uma maneira de armazenar e recuperar informações de banco de dados que seja conveniente e eficiente (KHAN *et al.*, 2010). Os SGBDs de alguma maneira, permitem o acesso aos dados, autorizar o acesso às informações a múltiplos usuários e manipular os dados presentes no banco de dados (inserção, supressão e modificação).

Um SGBD é um *software* complexo, otimizado para alguns tipos de processamento (por exemplo, responder a consultas complexas ou tratar várias requisições concorrentes), e seu desempenho pode não ser adequado para determinados aplicativos especializados (RAMAKRISHNAN; GEHRKE, 2008), fato pelo qual os SGBDs se adequam aos tipos de bancos de dados. Quando o usuário for definir como implantar a nossa aplicação, é necessário definir antes tipo de banco de dado que será utilizado, atualmente os banco de dados estão categorizados em três tipos: Banco de Dados Relacional, NoSQL e NewSQL. Em seguida, serão discutidos em detalhes cada um desses tipos de banco de dados.

2.1.1 Banco de Dados Relacional

Um banco de dados relacional modela e armazena dados em estruturas chamadas tabelas, esta estrutura é composta por colunas e linhas, acessada em forma de registros ou tuplas. Um banco de dados relacional contém, geralmente, muitas relações, com suas tuplas relacionadas de várias maneiras entre si (ELMASRI; NAVATHE; MORAIS, 2005). Sua linguagem usual é a *Structured Query Language* (SQL), que é uma linguagem padrão para bancos de dados relacionais.

Um banco de dados relacional é representado por um modelo relacional como uma coleção de relações. O modelo relacional se dedica ao exame de três aspectos principais de dados: a estrutura de dados, a manipulação de dados e a integridade de dados (DATE, 2004), no qual os componentes são bastante interligados de tal modo que se remover um dos componentes o modelo inteiro se torna ineficiênte ou destruída. Na terminologia do modelo relacional formal, uma linha é chamada **tupla**, um cabeçalho de **coluna** é conhecido como **atributo**, e a tabela é chamada **relação** (ELMASRI; NAVATHE; MORAIS, 2005).

Um esquema de um banco de dados representa, visualmente, a estrutura de um banco de dados e contém regras que gerencia o banco de dados e sua organização de dados. Um esquema de um banco de dados relacional S é um conjunto de esquemas de relação S = *R1, R2,..., Rm* e um conjunto de Restrições de Integridade RI. Um estado de um banco de dados relacional DB de S é um conjunto dos estados da relação DB = *r1, r2, ..., rm*, de forma que cada *ri* seja um estado de R e de maneira que os estados da relação *ri* satisfaçam as RI especificadas (ELMASRI; NAVATHE; MORAIS, 2005). Algumas aplicações que suporta SQL para suprir suas necessidades, criam extensões SQL. SchemaSQL é uma dessas extensões para SQL que permite manipulação uniforme de dados e suporta: (a) interoperabilidade em um sistema mutli-bancos e (b) especificação fácil de várias consultas complexas e cálculos decorrentes de aplicativos de armazenamento de dados (LAKSHMANAN; SADRI; SUBRAMANIAN, 1999).

2.1.2 NoSQL

Os sistemas de bancos de dados NoSQL surgiram como uma solução para a questão da escalabilidade no armazenamento e processamento de grandes volumes de dados na *Web* 2.0. No início, grandes empresas enfrentando esse tipo de problema, criaram suas próprias soluções, e publicaram alguns artigos científicos descrevendo diversas soluções ligadas ao gerenciamento de dados distribuído em larga escala, mas sem usar ainda o nome NoSQL (DIANA; GEROSA,

2010). Com avanço dessa tecnologia, sua adoção na indústria e academia, surgiu o nome NoSQL.

Segundo Moniruzzaman e Hossain (2013), NoSQL, refere-se a um grupo eclético e cada vez mais familiar de sistemas de gerenciamento de dados não-ralacionais; onde o esquema de armazenamento de dados não são principalmente em tabelas e geralmente não usam SQL para manipulação de dados . Os SGBDs NoSQL são úteis quando se trabalha com uma enorme quantidade de dados e quando a natureza dos dados não requer um modelo relacional.

É certo que os sistemas de bancos de dados NoSQL foram implementados para suprir a ineficiência dos sistemas relacionais tradicionais que se encaixam apenas para pequenas ou frequentes transações de leitura / gravação ou para grandes transações por lotes com acessos raros de gravação e não para cargas de trabalho pesadas de leitura / gravação (o que é frequentemente o caso para esses serviços web em grande escala - como o Google, Amazon, Facebook, Yahoo etc) (TUDORICA; BUCUR, 2011). As cargas de trabalho destas organizações destacadas e outras, tiveram crescimento exponencial ao longo dos últimos anos, de modo que só SGBDs alternativos ao SGBD Relacional tradicional como NoSQL ou NewSQL possam lidar.

Apesar de sistemas NewSQL fornecer uma tecnologia evoluída, isto é, superam os limites convencionais de desempenho do SGBD relacional tradicional empregando recursos de estilo NoSQL, como armazenamento de dados orientado a coluna e arquiteturas distribuídas, ou empregando tecnologias como processamento na memória, multiprocessamento simétrico ou processamento massivamente paralelo (MONIRUZZAMAN; HOSSAIN, 2013), o NoSQL ainda é mais aceitável no mercado, dado que grandes organizações de Tecnologia de Informação usam SGBD NoSQL proprietário ou alugado.

2.1.3 NewSQL

A principal limitação dos SGBDs relacionais tradicionais é capacidade de dimensionar seus recursos para processar volume variável de cargas de trabalho concorrente de diferentes tipos de requisições num ambiente de enorme quantidade e variadade dados, pois para dar o suporte adequado a clientes, como por exemplo, uma taxa de transferência maior, exigia uma atualização para um servidor maior. Até recentemente, a implementação de uma arquitetura de dimensionamento requer um modelo de programação NoSQL ou dependia de replicação exagerada e fragmentada. Não houve soluções que forneçam semântica ACID completa. Essa tensão é o que inspirou o movimento NewSQL (KAUR; SACHDEVA, 2017), um sistema de gerenciamento de banco de dados relacional escalável para OLTP que fornece desempenho

escalável como dos sistemas NoSQL para cargas de trabalho de leitura e gravação, além de manter as garantias ACID de um sistema de banco de dados tradicional (MONIRUZZAMAN, 2014).

Os sistemas NewSQL recentemente propostos visam alcançar a alta escalabilidade e disponibilidade do mesmo nível que o NoSQL, preservando as propriedades ACID para transações e funcionalidades complexas de bancos de dados relacionais. Os sistemas de banco de dados NewSQL são apropriados em aplicativos onde o SGBD tradicional foi usado, mas exigindo escalabilidade adicional e aprimoramento de desempenho (YUAN *et al.*, 2015). Permite que os aplicativos executem um grande número de transações simultâneas para ingerir novas informações e modificar o estado do banco de dados usando o SQL ao invés de uma API (Interface de Programação de Aplicação) proprietária (PAVLO; ASLETT, 2016).

As aplicações visadas pelos SGBDs do NewSQL, por outro lado, são caracterizadas como a execução de transações de leitura e gravação que (1) são de curta duração, (2) atinja um pequeno subconjunto de dados usando pesquisas de índice (i.e., sem varreduras de tabela completa ou junções distribuídas grandes) e (3) são repetitivas (ou seja, executando as mesmas consultas com diferentes entradas). Outros optam por um uma caracterização mais restrita onde a implementação de um sistema NewSQL tem que usar (1) um esquema de controle de simultaneidade sem bloqueio e (2) uma arquitetura distribuída não compartilhada (PAVLO; ASLETT, 2016).

Os sistemas de bancos de dados NewSQL delibera problemas de escalabilidade, desempenho e disponibilidade que geralmente se encontra nos istemas tradicional ou seja, banco de dados Relacional e fornecem capacidade de consulta SQL. A maioria deles segue o modelo de dados relacionais. O NewSQL deve ser considerado como uma alternativa ao NoSQL ou banco de dados relacional clássico para novos aplicativos OLTP (KAUR; SACHDEVA, 2017). A seguir, a lista de características técnicas do NewSQL:

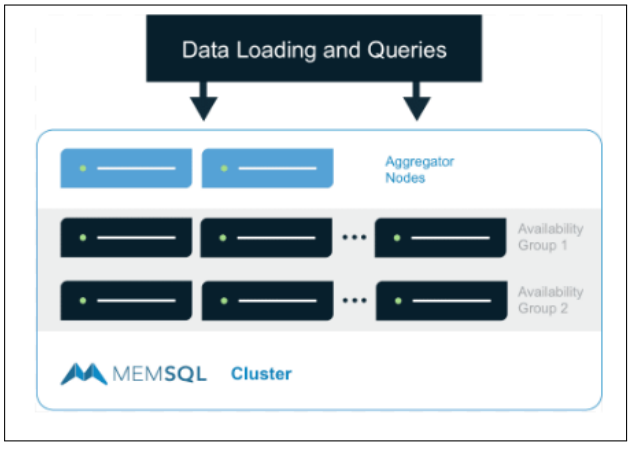
- SQL como o principal mecanismo de interação de aplicativos;
- Suporte ACID para transações;
- Um mecanismo de controle de simultaneidade não bloqueável, portanto as leituras em tempo real não entrarão em conflito com as escritas;
- Uma arquitetura que oferece um desempenho por nó muito maior que o disponível nas soluções SGBDs tradicionais;
- Uma arquitetura de escala, não compartilhada, capaz de funcionar em um grande número de nós sem sofrerem estrangulamentos.

NewSQL como um SGBD que visa suprir algumas limitações de NoSQL e SQL principalmente, leva em consideração a uma certa ordem de categorização baseados nos aspectos mais destacados de sua implementação. Segundo Pavlo e Aslett (2016), as três categorias que melhor representam os sistemas NewSQL são:

1. sistemas inovadores construídos a partir do zero usando uma nova arquitetura;
2. *middleware* que re-implementam a mesma infra-estrutura que foi desenvolvida na década de 2000 pelo Google e outros; e
3. ofertas de banco de dados como serviço de provedores de computação em nuvem que também são baseadas em novas arquiteturas.

Os SGBDs NewSQL não necessariamente apresentam a mesma arquitetura, dependendo da sua categoria, pode apresentar arquitetura semelhante de um SGBD da mesma categoria e diferente, caso não sejam da mesma categoria. Embora os sistemas NewSQL variem muito em suas arquiteturas internas, os dois recursos distintivos comuns entre eles são, todos eles suportam o modelo de dados relacionais e usam o SQL como sua principal interface (MONIRUZZAMAN, 2014). A Figura 1 mostra a arquitetura de um dos banco de dados NewSQL popular, o MemSQL, onde as cargas de trabalho de aplicativos podem ser divididas entre grupos, em que um grupo atende ao Aplicativo A, outro grupo atende o Aplicativo B e assim por diante.

Figura 1 – Arquitetura de danco de dados NewSQL popular, MemSQL



Fonte – MemSQL (2015)

A categoria “Nova arquitetura” geralmente contém sistemas NewSQL mais interessantes, fato pelo qual são projetados e implementados a partir do zero, isto é, não foi construído a partir de um outro sistema existente. Todos os SGBDs nesta categoria são baseados em arquiteturas distribuídas que operam em recursos não compartilhado e contêm componentes para suportar o controle de concorrência de vários nós, tolerância a falhas através de replicação, controle de

fluxo e processamento de consulta distribuída (PAVLO; ASLETT, 2016). Essa categoria fornece muitas vantagens devido a possibilidade que os desenvolvedores têm de projetar sua arquitetura e implementar sem preocupar com detalhes herdado de um sistema legado. Segundo (PAVLO; ASLETT, 2016) isso significa que o SGBD é responsável por distribuir o banco de dados em seus recursos com um mecanismo personalizado, ao invés de depender de recurso como um sistema de arquivos distribuído *off-the-shelf* (por exemplo, HDFS) ou estrutura de armazenamento (por exemplo, Apache Ignite).

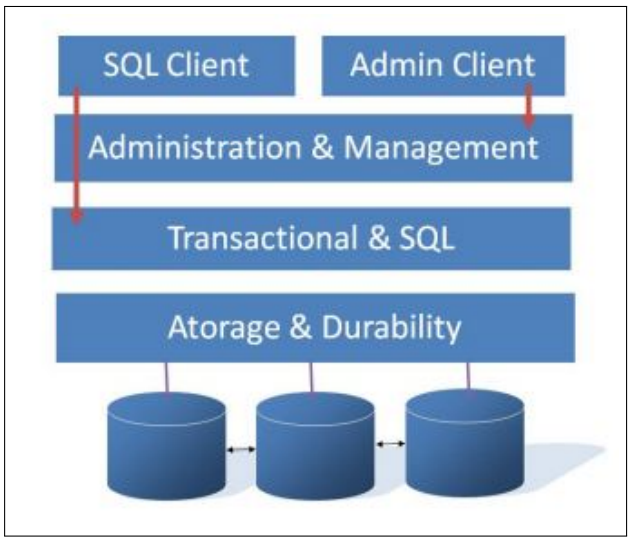
2.1.3.1 NuoDB

O NuoDB é um banco de dados SQL distribuído, *peer to peer*, que fornece um serviço de banco de dados na memória com transações ACID, aparece para o desenvolvedor e o operador como um único sistema lógico Kaur e Sachdeva (2017). Na prática, um banco de dados NuoDB pode ser executado em vários locais com *hosts* adicionados e removidos de acordo com a demanda. Assim como outros banco de dados NewSQL possui a capacidade de dimensionar cargas de trabalho de processamento de transações *on-line*, de modo que os sistemas legados não conseguem e é projetado para ambientes de nuvem e de contêiner. Opera sob recursos não compartilhado, possui tolerância a falhas através de mecanismo de replicação e processa de consultas de forma distribuída.

Sua arquitetura *peer to peer* garante que os serviços de banco de dados possam ser distribuídos nativamente em vários nós, como centros de dados e até mesmo nuvens, sem complexidade, sem mais despesas e nem *software* adicional que os bancos de dados relacionais tradicionais exigem. Devido a sua correspondência, qualquer par pode falhar ou ser encerrado a qualquer momento sem perder o banco de dados como um todo, dado que não há coordenador ou outro ponto único para falha, pois sua arquitetura *peer to peer* consiste em processos de *peer* independentes e capazes de realizar as mesmas tarefas. A Figura 2 mostra a arquitetura do banco de dados NuoDB.

Segundo Sila (2015), sua arquitetura consiste em três camadas: administrativa (aplicação), transacional e memória. A camada administrativa representa a entrada do usuário através das interfaces, denominados de agentes. camada transacional contém as unidades de transação de processos e as unidades de memória são armazenadas na camada de memória.

Figura 2 – Arquitetura do banco de dados NuoDB



Fonte – Moniruzzaman (2014)

2.2 APRENDIZADO DE MÁQUINA

A Inteligência Artificial vem da interação homem-máquina, no qual as máquinas possuem capacidade de tomar decisões como seres humanos, isto é, decidir de forma inteligente. Segundo Monard e Baranauskas (2003), AM é uma área de Inteligência Artificial cujo objetivo é o desenvolvimento de técnicas computacionais sobre o aprendizado bem como a construção de sistemas capazes de adquirir conhecimento de forma automática. Um sistema de AM supervisionado é um programa (indutor) capaz de induzir uma descrição de um conceito (classificador) a partir de um conjunto de exemplos conhecidos e previamente rotulados com as suas respectivas classes (PRATI; BARANAUSKAS; MONARD, 2002), em outras palavras, ele toma decisões baseado em experiências acumuladas de solução bem sucedida de problemas anteriores.

No que se refere a eficiência de algoritmos de AM, é improvável que um único algoritmo tenha um bom desempenho para todos os problemas dessa natureza. Diferentes algoritmos de AM podem ser aplicados à uma mesma amostra de dados e alguns desses algoritmos podem apresentar melhor desempenho que outros (PRATI; BARANAUSKAS; MONARD, 2002). Portanto, é necessário avaliar conceitos de indução desses algoritmos em determinados problemas, para mensurar seus desempenho.

Levando em consideração a hierarquia, de acordo Monard e Baranauskas (2003), os sistemas de aprendizado podem ser classificados em duas grandes categorias:

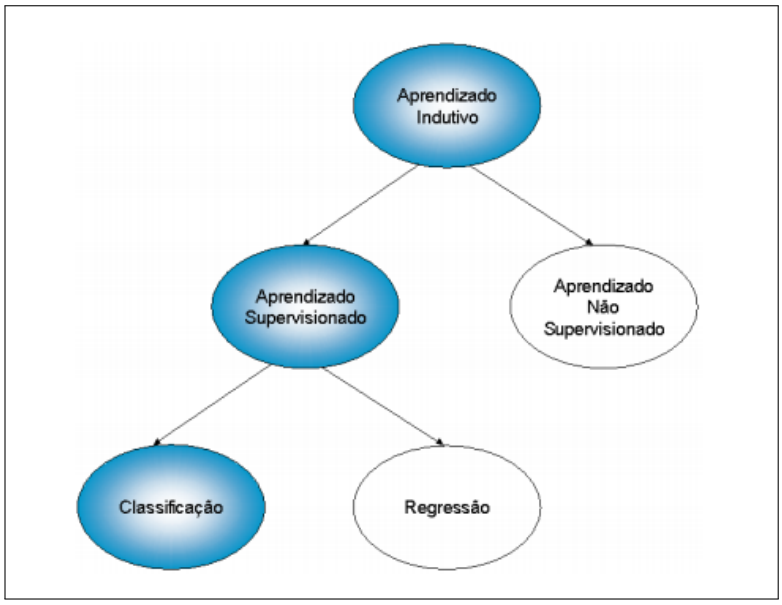
1. Sistemas tipo caixa-preta que desenvolvem sua própria representação do conceito, isto é, sua representação interna pode não ser facilmente interpretada por humanos e não

fornecem nem esclarecimento, nem explicação do processo de reconhecimento;

2. Sistemas orientados a conhecimento que objetivam a criação de estruturas simbólicas que sejam compreensíveis por humanos.

A Figura 3 exiba a hierarquia de aprendizado, onde também é possível ver aprendizados indutivas da mesma camada.

Figura 3 – A hierarquia do aprendizado



Fonte – Monard e Baranauskas (2003)

Pode-se constatar que os classificadores eram visto como caixa-preta e os algoritmos comportam de maneira única para cada entrada de dados, visto que para cada prescrição ou ramo da árvore, são necessárias informações que permitam proceder medidas acurada e interesse dessas regras.

2.3 COMENTÁRIO GERAL DO CAPÍTULO

Este capítulo apresentou os conceitos necessários que alicerçam este trabalho. Inicialmente foram definidos os aspectos importantes sobre banco de dados, onde se fez uma explanação de exemplos descritivos ilustrativos e ainda se enfatizou alguns detalhes importantes sobre SGBD. Em seguida foram explorados os conceitos e detalhes técnicos de cada um dos tipos de banco de dados vigente: banco de dados Relacional, destacaram as características essenciais que envolve seu modelo e sua esquema; NoSQL foi abordado assuntos como motivação da sua criação, suas características em geral, sua aplicabilidade e seu diferencial em relação de

banco de dados Relacionais; NewSQL, também foi ilustrado a motivação da sua criação, sua definição e ainda foram apresentados a definição de características específicas da qual um banco de dados NewSQL se enquadra e suas características técnicas, foi listada suas três diferentes categorias. Além disso, foram explorados detalhes importantes da sua arquitetura, onde foi feito a explanação da arquitetura de sistema de banco de dados NewSQL, o MemSQL e ainda foi estudada de forma resumida o conceito de um outro banco de dados popular NewSQL, o NuoDB. E por fim, foi abordado o conceito de AM, onde foi enfatizada sua definição, sua categorização e sua hierarquia.

3 TRABALHOS RELACIONADOS

3.1 INTRODUÇÃO

Este capítulo apresenta os principais trabalhos relacionados ao presente trabalho, isto é, os que propõem modelagens de desempenho para bancos de dados de modo geral, em especial o trabalho de Farias *et al.* (2017) no qual este trabalho é baseado. Esses trabalhos foram selecionados nos mais renomados plataformas de busca por trabalhos ciêntífico como, journal ScienceDirect, conferências IEEEXplore e ACM Digital Library, usando o critério de seleção com as palavras chaves *performance modeling*, *SLA* (Acordo de Nível de Serviço), *databases* e *TI* (Tecnologia de Informação) dos trabalhos publicados nos útimos 5 anos (i.e, a partir de 2014). Ao final deste capítulo será realizada uma análise comparativa entre os trabalhos expostos.

3.2 TRABALHOS SELECIONADOS

Esta seção procurou-se selecionar os trabalhos mais expressivos que fazem modelagem de desempenho e levam em consideração na modelagem as métricas de desempenho de seu interesse e que gera modelos de desempenho acurados.

3.2.1 Regression based performance modeling and provisioning for NoSQL cloud databases

Em Farias *et al.* (2017), os autores propõem uma abordagem de modelagem de desempenho baseada no SLA para sistemas de bancos de dados NoSQL. Seu trabalho leva em consideração cargas de trabalho heterogêneas e capazes de capturar os efeitos não-lineares dos aspectos de concorrência e distribuição sobre o desempenho. Para isso, usaram algoritmos de aprendizado de máquinas, como métodos de *Gradient Boosting Machine* e *Regressão Linear* capazes de modelar qualquer tipo de relações não lineares, e ainda usaram características lineares e quadráticas para que os modelos gerados por essas técnicas a partir do conjunto de treinamento tenham poder de generalização e extrapolação.

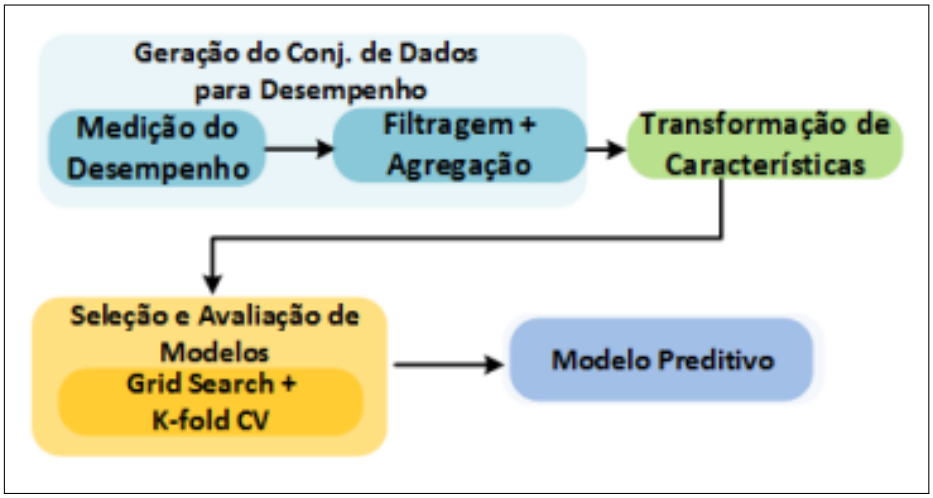
O trabalho também propõe uma estratégia de provisionamento elástico automático baseado em modelos de desempenho da fase anterior, que adiciona e remove recursos em tempo de execução de acordo com as características da carga de trabalho corrente. Adicionalmente, a abordagem de provisionamento automático faz proveito do gerenciamento da incerteza produzidos pelos modelos de desempenho.

O trabalho foi dividido em duas abordagens: **Modelagem de Desempenho** e **Provisionamento Elástico**.

3.2.1.1 Modelagem de Desempenho

Onde construíram um modelo ou função de desempenho que recebe as características da carga de trabalho e as características do *cluster* como entrada para estimar o desempenho do serviço nesse cenário por meio de algoritmos de aprendizado de máquina, dividido em quatro etapas ilustrada na Figura 4.

Figura 4 – Visão geral da modelagem do desempenho



Fonte – Farias *et al.* (2017)

3.2.1.2 Provisionamento Elástico

Monitora a carga de trabalho e calcula, dinâmica e automaticamente, a alocação ótima de acordo com as características da carga de trabalho corrente ao longo da execução do sistema.

Na fase de **monitoramento** obtenham-se informações sobre a carga de trabalho de entrada no sistema periodicamente. Considerou-se um intervalo de monitoramento de 1 segundo.

Na fase de **provisionamento**, com os dados provenientes do sistema de monitoramento, a estratégia de provisionamento verifica periodicamente se alguma ação de provisionamento é necessária e, em caso positivo, calcula qual deve ser a nova alocação de recursos. O período em que esse processo acontece é definido como ciclo de provisionamento. Ainda nessa fase, faz-se gerenciamento de incertezas de provisionamento com modelo de incerteza

construído na fase de modelagem de desempenho para enriquecer abordagem proposta.

Embora esta abordagem de modelagem de desempenho possui alta capacidade de generalização dos modelos de desempenho e incerteza, da qual considero um modelo de desempenho acurado, sua abordagem de provisionamento elástico carece de técnicas de predição capazes de antecipar o comportamento de carga do sistema e, assim, decidir pelas ações de elasticidade.

3.2.2 TIER : Table Index Evaluator and Recommender - A Proposed Model to Improve Transaction Performance in Distributed Heterogeneous Database

TIER: é uma ferramenta proposta por (NAIK, 2017) para melhorar desempenho de transações de banco de dados distribuídas heterogêneas e contribuirá na resolução do problema de seleção de índice. O modelo é proposto com base em pesquisa e revisão de literatura dos métodos existentes. A pesquisa foi realizada com pessoas experientes da indústria de TI para descobrir a viabilidade do modelo proposto.

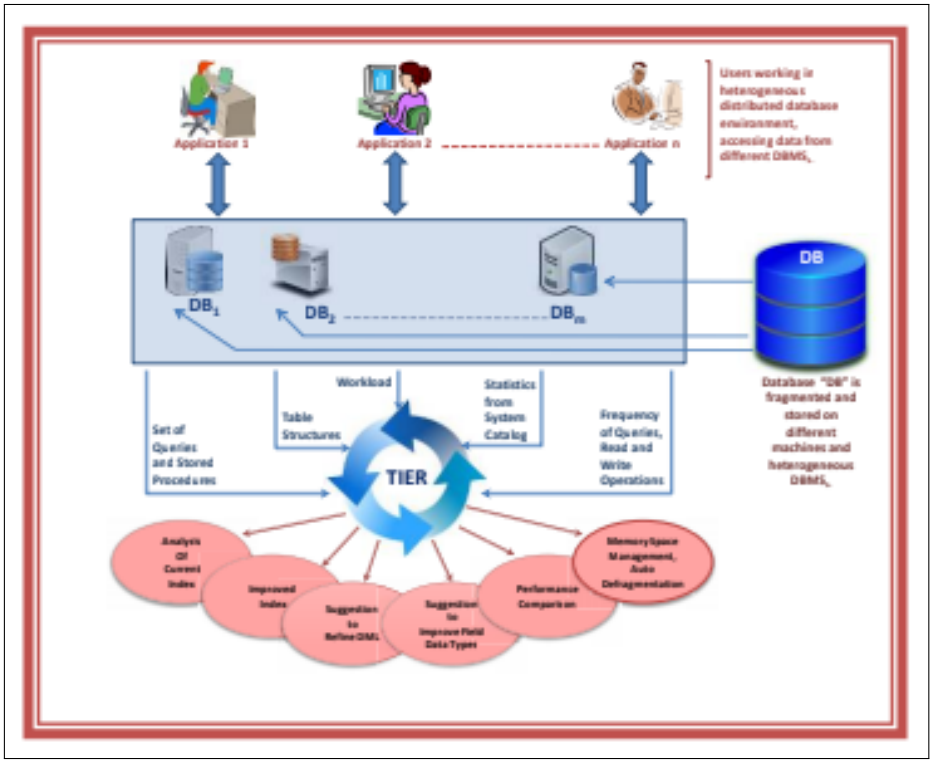
Por índice ser uma estrutura física, ele ocupa a memória e, uma vez definido, requer monitoramento constante. A mudança na carga de trabalho afeta o desempenho do índice. Portanto, o banco de dados precisa de uma ferramenta que analise os índices existentes, crie um novo índice ou reconstrua o índice existente e, se necessário (NAIK, 2017). O autor alegou que houve uma carência de eficiência nas ferramentas existentes, no que diz respeito principalmente as consultas agregadas, subconsultas e outras consultas complexas, por isso, propuseram um Avaliador e Recomendador de Índice da Tabela (TIER). A Figura 5 ilustra o modelo proposto, o TIER, onde é possível ver sua composição e modo de interação dos seus componentes.

O TIER analisará e sugerirá a inserção de melhores índices para as tabelas de banco de dados que são armazenadas em ambiente heterogêneo distribuído, no qual utilizará várias entradas do banco de dados e, usando um algoritmo, avaliará os índices existentes e sugerirá novos índices. O modelo oferecerá alguns outros benefício a mais, como sugestão de melhoria dos tipos de dados e restrições; e desfragmentação de índice e ainda permitirá a comparação de desempenho com o índice existente após a implementação do índice revisado.

3.2.3 Cloud Resource Autoscaling System based on Hidden Markov Model (HMM)

Em (NIKRAVESH; AJILA; LUNG, 2014) os autores propõem um sistema de dimensionamento automático baseado no Modelo de *Hidden Markov* (HMM), uma ferramenta

Figura 5 – TIER: Avaliador e Recomendador de Índices de Tabela - O Proposto Modelo



Fonte – Naik (2017)

onipresente para modelagem de dados temporais em séries; são usados para capturar e modelar observações estocásticas. Este trabalho é uma abordagem para lidar com o comércio de custo-desempenho que ajusta automaticamente os recursos do aplicativo com base em sua carga.

Considerando três partes interessadas no ambiente da computação em nuvem, isto é , provedor, cliente e usuário, o presente trabalho analisa o problema do ponto de vista do Cliente, no qual por um lado, tenta minimizar seu custo, diminuindo seus recursos e, por outro lado, é obrigado a manter o SLA com os usuários.

Para criar e avaliar o modelo HMM, foi utilizado o software Weka, uma ferramenta de aprendizagem de máquinas para tarefas de mineração de dados (HALL *et al.*, 2009), é importante notar que na versão padrão do Weka o algoritmo HMM não está incorporado, por isso, usaram a extensão HMMWeka para seus experimentos. Além disso, para gerar dados históricos, foi utilizado o *benchmark* TPC-W como o gerador de carga. O TPC-W é uma especificação de referência que é comumente usada para provisionamento de recursos, escalabilidade e planejamento de capacidade para sites tradicionais de comércio eletrônico. Também realizaram experimento na infraestrutura Amazon EC2 para avaliar modelo. Os resultados mostram que o HMM pode gerar ações de escala corretas em 97 porcento do tempo. A utilização da CPU, a taxa de transferência e o tempo de resposta são considerados como métricas de desempenho

nesta experiência.

Embora este sistema de modelagem de desempenho baseado na elasticidade seja proativo e apresenta uma percentagem bastante alta de índice de ações de escala corretas, mas é necessária uma experiência mais detalhada nas nuvens reais como Amazon e outras para confirmar completamente esta conclusão.

3.2.4 A Database Performance Polynomial Multiple Regression Model

Em (NOWOSIELSKI; KOWALSKI; KULCZYCKI, 2017), os autores desenvolveram um modelo matemático de desempenho de banco de dados Orientado à Colunas (COBD). As variáveis ilustrativas obrigatórias do modelo consiste em um número de colunas e um número diferente de valores presentes em um banco de dados e solicitações emitidas nele. Os coeficientes do modelo são otimizados pelo Algoritmo de Polinização por Flores (FPA). O modelo é baseado em método de regressão linear com parâmetros opcionais. O algoritmo é executado com inúmeras combinações de parâmetros possíveis; os resultados e potenciais obstáculos são discutidos e abordagens de modelagem alternativas são mencionadas.

O conjunto de variáveis ilustrativas inclui informações sobre o percentual de participação de diferentes tipos de operações do banco de dados. Os dados foram coletados em duas fases:

1. A primeira fase coletou os dados necessários para avaliar o formato da função para fatores específicos, onde o *benchmark* de banco de dados foi executado inúmeras vezes com configurações diferentes, mas regulares, a fim de isolar os dados sob a influência de fatores específicos no desempenho geral. Em seguida, os dados referente a fatores específicos foi visualizado com um gráfico de dispersão e curvas em um espaço tridimensional. Formas de função para fatores específicos são reconhecidas por uma avaliação manual de gráfico visual. O modelo foi dividido em duas variantes neste estágio. A segunda variante enriquece a primeira, incluindo uma informação sobre a proporção entre as operações do banco de dados. A primeira variante ignora esse recurso, enquanto o segundo o utiliza, na forma de componentes variáveis composicionais fornecidos diretamente à equação de regressão. Ambas as variantes são comparadas entre si em termos de precisão do modelo.
2. Enquanto que a segunda fase aumentou o valor estatístico dos dados de entrada do modelo, onde o *benchmarking* usou valores aleatórios obtidos da distribuição uniforme como variáveis de entrada.

A divisão da coleta de dados empíricos nestes dois estágios teve como objetivo garantir a alta qualidade dos dados de entrada. Os coeficientes são otimizados para o erro mínimo, ou seja, a maior precisão. Este modelo foi avaliado como polinômio de terceira ordem, o que resultou na equação de regressão com 7 coeficientes. Uma variante com taxas de operação aumentou o número de coeficientes para 10.

Esperam-se que no futuro, o modelo possa se beneficiar da experimentação de outros algoritmos metaheurísticos, como o Algoritmo Krill Herd. A otimização numérica é também um dos aparelhos mais típicos de algoritmos evolutivos e genéticos. Uma transformação de valor de composição, como Transformação de relação Aditivo / Centralizado / Log Isométrico (ALR, CLR, ILR) deve ser usada como variável composicional de operações. Isso pode melhorar a precisão do modelo e permitir a análise sem comparar as duas variantes do modelo.

3.3 DISCUSSÃO

Nesta seção, é exposto um quadro comparativo dos trabalhos apresentado na Seção 3.2 que fazem modelagem de desempenho para Bancos de Dados, seja NoSQL ou Relacional. Foram destacados os tipos de banco de dados utilizados, a estratégia de modelagem e análise de cada trabalho tratado.

Quadro 1 – Ilustração do resumo comparativo com as características de cada trabalho relacionado e da presente proposta

Trabalho	**Abordagem**	**Bancos de Dados**	**Estratégia de Modelagem**	**Análise de Cargas de Trabalhos**
Farias *et al.* (2017)	Modelagem do Desempenho e Elasticidade	NoSQL	Aprendizado	Heterogêneas
Naik (2017)	Modelagem do Desempenho	Distribuído	Não se aplica	Heterogêneas
Nikravesh, Ajila e Lung (2014)	Modelagem do Desempenho e Elasticidade	NoSQL	Aprendizado e analiico	Heterogêneas
Nowosielski, Kowalski e Kulczycki (2017)	Modelagem do Desempenho	NoSQL	Aprendizado	Heterogêneas
Abordagem Proposta	Modelagem do Desempenho	Relacional Distribuído (NewSQL)	Aprendizado	Heterogêneas

Fonte – Elaborado pelo autor

Na Seção 3.2 foram destacados os pontos fortes e limitações de cada trabalho. Todos

estes trabalhos usam alguma abordagem de modelagem do desempenho e são razoavelmente boas, dado que suas técnicas de modelagem seguiram umas certas regras e padrões matemáticos ou estatísticos e apresentam resultados confiáveis do ponto de vista de suas experimentações. Os pontos negativos detectados são correlação a carga de trabalho, pois não tratam cargas de trabalho heterogêneas ou o tratamento é bastante simples, exceto o trabalho do Farias *et al.* (2017).

3.4 COMENTÁRIO GERAL DO CAPÍTULO

Neste capítulo foram apresentados os principais trabalhos relacionados ao tema deste presente trabalho. Foram expostas as características e particularidades de cada um desses trabalhos, seus pontos fortes e suas limitações. Foi foita comparação deste trabalho com os presentes trabalhos relacionados destacados neste capítulo. Ao analisar esses trabalhos, vimos que maioria deles restringem modelagem de desempenho para sistemas de Bancos de Dados NoSQL, não para sistemas de banco de dados em geral, o que justifica o intuito da nossa proposta. E também muitos desses trabalhos não tratam cargas de trabalhos heterogêneas.

4 UMA ABORDAGEM PARA MODELAGEM DE DESEMPENHO PARA SISTEMAS DE BANCO DE DADOS NEWSQL E COMPARAÇÃO DE MODELOS GERADOS COM DIFERENTES TÉCNICAS DE APRENDIZADO DE MÁQUINA

4.1 INTRODUÇÃO

Por mais de quarenta anos, os bancos de dados relacionais têm sido o principal modelo de armazenamento, recuperação e gerenciamento de dados. No entanto, devido às crescentes necessidades de escalabilidade e desempenho, surgiram sistemas alternativos, a tecnologia NewSQL (KAUR; SACHDEVA, 2017), o mais recente e NoSQL. Um dos principais avanços na resolução do problema de escalabilidade e de desempenho foi o surgimento dessas tecnologias. Hoje, o SGBD Relacional clássico é complementado por esses sistemas de gerenciamento de dados, especialmente projetados para lidar com o volume, a variedade, a velocidade e a variabilidade das coleções de grandes quantidades de dados (MONIRUZZAMAN, 2014). NewSQL é uma classe de modernos SGBD Relacional que fornece o mesmo desempenho escalável dos sistemas NoSQL para cargas de trabalho de leitura e gravação ao OLTP enquanto ainda mantêm as garantias ACID de um SGBD tradicional.

Os sistemas de bancos de dados NewSQL, de modo semelhante os NoSQL, estão integrando ambientes distribuídos de processamento de dados que executam cargas de trabalhos concorrentes e heterogêneas compostas por requisições. Ao mesmo tempo, esses sistemas precisam satisfazer as expectativas de desempenho definidas no Acordo de Nível de Serviço (SLA). Nesse ambiente, a previsibilidade de desempenho pode ajudar a evitar potenciais violações do SLA, i.e., a habilidade de estimar o impacto da execução concorrente de requisições em uma carga de trabalho em execução (FARIAS *et al.*, 2017).

Este trabalho é baseado no trabalho de Farias *et al.* (2017), mas voltado para sistemas de Banco de Dados NewSQL que propõe uma abordagem para modelagem de desempenho baseada no SLA para sistemas de bancos de dados NewSQL. Essa abordagem leva em consideração cargas de trabalho heterogêneas e é capaz de capturar os efeitos não-lineares dos aspectos de concorrência e distribuição sobre o desempenho.

Este trabalho ainda faz uma comparação do desempenho entre os modelos gerados usando métodos de aprendizado de máquina que o Farias *et al.* (2017) usou, *Regressão Linear* e *Gradient Boosting Machine*, e os modelos gerados usando métodos de aprendizado de máquinas, *Extremely Randomized Trees* e *AdaBoost*, (PEDREGOSA *et al.*, 2011).

4.2 VISÃO GERAL

Este trabalho tem o desafio de lidar com cargas de trabalhos que possui tipos de requisições diferentes (leitura e escrita) e complexidades diferentes (consultas por igualdade, consultas por intervalo, atualizações e inserções). São agrupadas manualmente as requisições de cargas de trabalho em classes. Cada classe contém requisições que são do mesmo tipo (leitura ou escrita) e possuem complexidades semelhantes. Foi usado o *benchmark* YCSB (YCSB, 2015) que possui drivers específicos para sistemas de banco dados NewSQL para gerar cargas de trabalho.

O YCSB produz quatro tipos de requisições: *read*, *scan*, *update* e *insert*. Desse modo, foi executado um agrupamento manual das requisições do YCSB em quatro classes, de modo que cada classe corresponde a um tipo de requisição. Note que, apesar deste trabalho considerar uma carga de trabalho específica, as discussões sobre carga de trabalho ao longo deste trabalho podem ser entendidas de forma mais genérica. Este trabalho foi dividido em duas abordagens:

Modelagem do desempenho: foi contruído um modelo ou função de desempenho que recebe as características das cargas de trabalho e as características do servidor como entrada para estimar o desempenho do serviço nesse cenário por meio de algoritmos de aprendizado de máquina.

Procedimento de comparação do desempenho de modelos gerados: foi feita comparação do desempenho entre modelos gerados usando métodos de aprendizado de máquina que foram utilizados no trabalho de Farias *et al.* (2017), com os modelos gerados usando outros métodos de aprendizado de máquinas.

4.3 MODELAGEM DE DESEMPENHO

Nesta fase, foi construído um modelo de desempenho que recebe as características da carga de trabalho e do servidor e estima o desempenho do serviço nesse cenário por meio de algoritmos de aprendizado de máquina. Este trabalho utiliza três métricas de desempenho: **vazão** (quantidade de requisições executadas por segundo), **tempo de resposta médio** (tempo de resposta médio das requisições executadas a cada segundo) e **violações por segundo** (quantidade de requisições cujo tempo de resposta ultrapassa limite pré-definido no SLA). Estas métricas são intuitivas do ponto de vista dos *stakeholders* e refletem, de um modo geral, a experiência de uso da aplicação pelo usuário.

As características (*features*) extraídas da carga de trabalho são relativas às classes da carga de trabalho (*read*, *scan*, *insert* e *update*), foi extraída uma característica para cada classe (Seção 4.3.1.2) que representa o volume de requisições dessa classe que é gerado pelo *benchmark* YCSB. O modelo do desempenho considera o tamanho do servidor, pois tamanho maior de servidor indica maior poder computacional e, consequentemente, indica melhora no desempenho.

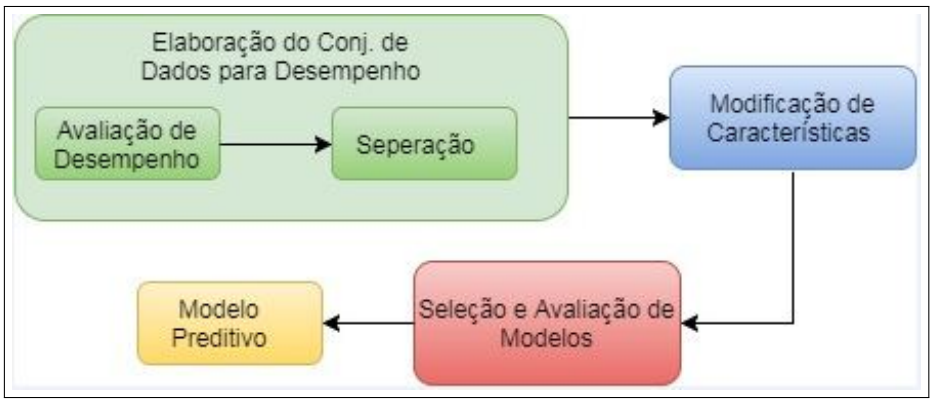
Este trabalho tem como objetivo ser genérico em relação ao SGBD utilizado. Sistemas NewSQL, assim como NoSQL são projetados para executar de forma distribuída usando um grande *pool* de recursos. O modelo de particionamento e gerenciamento de dados não são mesmos em todos SGBD. Por isso, foi extraída apenas a caraterística do tamanho do servidor, visto que esta é uma das únicas características que são comuns a todos sistemas NewSQL e que tem grande impacto no desempenho.

Neste trabalho foram tratados três aspectos que contribuem para a alteração do desempenho de um sistemas de banco de dados NewSQL: Complexidade das consultas, Concorrência e Distribuição, mas ainda existe mais fatores que pode afetar o desempenho.

1. Complexidade das consultas. Cada requisição exige um volume diferente de operações de disco. Isso faz com que, para retornar o resultado em tempo hábil, uma classe de requisições demande mais recursos computacionais do que outra classe;
2. Concorrência. A execução concorrente de requisições pode causar uma interferência por contenção de *lock* Mozafari *et al.* (2013) o que acarreta na degradação do desempenho por esperas e *deadlocks* entre as classes de requisições da carga de trabalho; e
3. Distribuição. Muitos sistemas de bancos NewSQL empregam estratégias de replicação para aprimorar a disponibilidade e confiabilidade. Contudo, isso implica no custo adicional de manter a consistência. Em geral, estratégias de consistência usam *locks* distribuídos o que causa esperas e *deadlocks* distribuídos.

Particularmente, o problema de inferir uma métrica de desempenho a partir das características da carga de trabalho e do tamanho do servidor é tratado como um problema de regressão, pois a variável alvo (métrica) a ser predita é uma variável contínua. O conjunto de dados de treinamento das técnicas de aprendizado de máquina é gerado em um ambiente de teste controlado, onde é possível executar várias configurações de cargas de trabalho sob demanda, o que caracteriza a abordagem para modelagem de desempenho como *offline*. Esta abordagem é dividida em três grandes etapas como ilustrado na Figura 6.

Figura 6 – Arquitetura de modelagem do desempenho



Fonte – Elaborado pelo autor

4.3.1 Elaboração do Conjunto de Dados de Desempenho

Essa etapa visa produzir um conjunto treinamento base para ser usado posteriormente em técnicas de aprendizado de máquina, pois técnicas de aprendizado de máquina necessitam de um conjunto de observações (i.e., conjunto de treinamento) de um determinado fenômeno para construir um modelo preditivo. Conjuntos de treinamento com pequena quantidade de amostras ou que apresentem intervalos limitados de características podem gerar modelos preditores com baixo poder de generalização e extrapolação. Além disso, um conjunto de treinamento de desempenho pode conter ruído, o que induz as técnicas de aprendizado de máquina a criar modelos preditivos menos acurados.

A geração desse conjunto de dados é dividida em duas subetapas: (1) Avaliação do Desempenho e (2) Separação:

4.3.1.1 Avaliação do Desempenho

Foram executados experimentos iniciais para medir o desempenho do SGBD sob diversos cenários, em um ambiente de teste separado do sistema de produção, de modo *offline*, para entender os impactos das características da carga de trabalho e do tamanho do servidor sobre o desempenho, utilizando o *benchmark* YCSB.

Foram selecionados os parâmetros que mais contribuem para a variação do desempenho que apresentam dinâmicas similares a cargas de trabalho reais, pois o YCSB oferece vários parâmetros que permitem mudar o padrão de acesso, o volume e a composição da carga de trabalho. Os parâmetros variados foram os seguintes: *target*, um inteiro que representa a quantidade de requisições por segundo geradas pelo *benchmark* e *mix*, um vetor [P_{read}, P_{scan},

P_{update}, P_{insert}] onde cada elemento representa a porcentagem total de requisições de cada tipo de operação. O elemento P_{read} representa a porcentagem de requisições de leitura por igualdade, P_{scan} representa a porcentagem de requisições de leitura por intervalo, P_{update} representa a porcentagem de requisições de atualização e P_{insert} representa a porcentagem de requisições de inserção. Note que *target* e *mix* afetam de forma direta o volume e a composição da carga de trabalho.

Deste modo, é necessário definir o conjunto de parâmetros relativos à carga de trabalho e ao tamanho do servidor que serão empregados nos testes. Vale lembrar que o intervalo usado para teste de cada parâmetro impacta diretamente na capacidade de generalização do modelo gerado. Por exemplo, no caso dos experimentos serem realizados com parâmetros *target* cujos valores pertençam ao intervalo (5000,7000), espera-se que o modelo seja menos acurado para valores de *target* menores que 5000 e maiores que 7000. O quadro a seguir mostra os valores de teste selecionados para cada parâmetro.

Quadro 2 – Quadro de valores de cada parâmetro para experimentos

Categoria	**Parâmetro**	**Valores**
Carga de trabalho	P_{read},	0%, 20%, 40%, 60%, 80% 100%
	P_{scan},	0%, 20%, 40%, 60%, 80% 100%
	P_{insert},	0%, 20%
	P_{update},	0%, 20%
	target	1000, 2000, 3000, 4000, 5000, 6000, 7000, 8000, 9000, 10000
Servidor	tamanho	1

Fonte – Elaborado pelo autor

Foi executado um experimento para cada combinação de parâmetros do Quadro 2. Note que nem todas as combinações são possíveis pois, por se tratarem de valores percentuais, $P_{read} + P_{scan} + P_{update} + P_{insert} = 1$. Dessa forma, foi adotado um total de 20 combinações de $mix = [P_{read}, P_{scan}, P_{update}, P_{insert}]$. Além disso, existem 10 possibilidades para *target* e 1 possibilidades para tamanho do servidor. Logo, tem-se o total 20x10x1 = 200 experimentos.

Em cada experimento, é gerado um arquivo de *log* que contém informações relativas às requisições processadas pelo sistema. A cada segundo foram coletadas as métricas de desempenho para a carga de trabalho:

1. O número de requisições executadas (vazão) no último segundo;
2. O tempo de resposta médio das requisições executadas no último segundo; e
3. A quantidade de violações de SLA por segundo.

4.3.1.2 Separação

Devido ao grande volume de informação, os dados brutos contidos no *log* não podem ser utilizados diretamente para realizar as análises preditivas pelos métodos de regressão, visto que foram 200 experimentos e a cada segundo uma entrada é gravada no *log*. Por isso, os *logs* gerados são agregados de forma a produzir um conjunto menor de dados. Estes dados foram formatados de tal sorte a serem utilizados como entrada nas técnicas de aprendizado de máquina. Estas técnicas requerem um conjunto de dados que é composto por duas partes: uma matriz de características contendo as amostras do fenômeno a ser modelado e um vetor de valores de saída referente a cada amostra da matriz de características.

Por o objetivo final dessa abordagem ser modelar o desempenho baseada nas três métricas listada na Seção 4.3.1.1, cada métrica foi tratada individualmente, i.e., foi gerado um conjunto de dados e um modelo preditivo para cada métrica. Para cada métrica de desempenho foi gerado um conjunto de dados (i.e., matriz de características e vetor de valores de saída). Cada experimento executado na Seção 4.3.1.1 é representado como uma linha da matriz de características, também chamada de vetor de características. Esse vetor é composto pelos parâmetros da carga de trabalho e tamanho do servidor utilizados em um dado experimento. Assim, este vetor contém 6 posições $< DB, t, t_{read}, t_{scan}, t_{update}, t_{insert} >$, onde DB é o tamanho do servidor no experimento, t é a quantidade de requisições alvo que são emitidas ao sistema (parâmetro *target*) e $tipo$ é a quantidade alvo de requisições de um tipo específico que são emitidas ao sistema, e.g, t_{read} é a quantidade alvo de requisições *read* por segundo, o que equivale ao parâmetro *target* multiplicado por P_{read}.

O valor de saída para esse vetor corresponde ao valor resumido de uma dada métrica de desempenho. Esse valor é obtido por meio da agregação dos dados filtrados do *log* desse experimento. Primeiramente, o primeiro minuto de experimento que corresponde ao aquecimento do *cache* é descartado. Em seguida, são removidos as medições que são maiores que o 80^{o} percentil e menores que o 20^{o} percentil. Isso serve para retirar o ruído causado por efeitos inesperados de escalonamento e interferência de outros clientes que rodam no mesmo servidor. Finalmente, o valor resumido corresponde à média das medidas do *log* filtrado.

4.3.2 Modificação das Características

É importante que os modelos gerados pelas técnicas de regressão a partir do conjunto de treinamento tenham poder de generalização. Essa é uma tarefa difícil, pois o modelo deve

capturar diversos aspectos como os citados na Seção 4.3 que afetam as métricas de desempenho e geram modelos de desempenho não-lineares.

De modo mais específico, Gray *et al.* (1996) sugerem de forma teórica que, para modelos de replicação mestre/escravo, o número de nós no *cluster* e o número de requisições impacta o desempenho de forma quadrática devido à contenção de *locks* gerada pela execução concorrente de requisições no mesmo SGBD relacional distribuído tradicionais. Esse trabalho estende esse conceito para SBGDs NewSQL, assim como NoSQL.

Os SBGDs NewSQL podem ter diversos modelos de distribuição de dados como replicação, fragmentação ou híbrido replicação/fragmentação. Vários desses modelos podem ser mais eficientes e escaláveis do que a replicação mestre/escravo. Assim, em geral, esses modelos de distribuição de dados são impactados quadraticamente ou menos em relação ao tamanho do servidor e de requisições emitidas. Logo, são necessárias técnicas de regressão que sejam capazes de capturar esses aspectos.

Essa etapa apresenta uma manipulação das características do conjunto de dados base para melhorar a qualidade dos modelos preditivos produzidos por técnicas de aprendizado de máquina, inclusive para que regressão linear modele as relações não-lineares, pois como citado anteriormente, o comportamento do desempenho pode estar correlacionado de forma não-linear com as características da carga de trabalho e do tamanho do servidor.

Por tal, este trabalho propõe duas manipulações de características que serão posteriormente testadas com uma variedade de métodos de regressão:

1. **Características lineares**: O conjunto de dados base gerado como descrito na Seção 4.3.1.2 é fornecido diretamente aos métodos de regressão. Essa transformação mais indicada para métodos que conseguem capturar superfícies arbitrárias. Para métodos de regressão linear, as características t, t_{read}, t_{scan}, t_{update}, t_{insert} contam para a contribuição de cada classe da carga de trabalho para o desempenho (aspecto 1, Seção 4.3). E *DB* representa o ganho ou perda de desempenho ao se aumentar ou diminuir o tamanho de um servidor.
2. **Características quadráticas**: Para modelar o efeitos quadráticos das características no valor de saída, o conjunto de dados base gerado na fase 4.3.1.2 foi modificado. Em complemento às características originais $< DB, t, t_{read}, t_{scan}, t_{update}, t_{insert} >$, foram adicionadas para cada amostra as seguintes características:
    - 15 características resultantes do produtos de todos os pares distintos das características de composição ($t, t_{read}, t_{scan}, t_{update}, t_{insert}$) incluindo cada termo ao quadrado:$<t^2$, t x t_{read}, t x t_{scan},...> . Essas características expressam a degradação do desempenho

causado pelos aspectos de concorrência (aspecto 2, Seção 4.3).

- 5 características resultantes dos produtos da característica *DB* com as características de composição t (t, t_{read}, t_{scan}, t_{update}, t_{insert}): <t, t_{read}, t_{scan}, t_{update}, t_{insert}>. Essas características representam o decaimento do desempenho por esperas e *deadlocks* distribuídos para cada classe da carga de trabalho. (aspecto 3, Seção 4.3)

4.3.3 Seleção e Avaliação de Modelos

Em geral, algoritmos de aprendizado de máquina têm vários parâmetros de entrada. Por exemplo, o método de regressão linear com regularização tem o parâmetro coeficiente de regularização para controlar a magnitude dos coeficientes. Desse modo, uma escolha inapropriada desses parâmetros pode acarretar na geração de modelos preditivos de baixa qualidade. O problema de escolher a melhor configuração de parâmetros de entrada é conhecido como otimização de parâmetros. Em um algoritmo de seleção de parâmetros é necessário escolher quais modelos preditivos são mais acurados.

Essa fase se concentra na escolha da melhor configuração de parâmetros e em mostrar como se mede a acurácia de um modelo preditivo. A saída dessa fase é o modelo de desempenho acurado para uma dada métrica e desempenho. A seguir é descrito a técnica usada para avaliar a acurácia do modelo e para a otimização de parâmetros.

4.3.3.1 Avaliação do Modelo

Para avaliar a acurácia de um modelo, foi utilizada a validação cruzada *K-fold* tradicional. Nessa técnica, o conjunto de dados é aleatoriamente dividido em k subconjuntos de tamanhos iguais. Assim, treinou-se os métodos de aprendizado de máquina com $k-1$ subconjuntos e o erro é calculado a partir do subconjunto restante. Esse processo é feito k vezes, uma vez para cada conjunto de treino composto por $k-1$ subconjuntos.

Como métrica de erro foi usado o coeficiente de determinação R^2 que é dado por:

$$R^2 = 1 - \frac{\sum_{i=1}^{N}(p_i - y_i)^2}{\sum_{i=1}^{N}(y_{media} - y_i)^2} \tag{4.1}$$

Onde yi é o valor de saída real da i-ésima observação do conjunto de teste e p_i denota o valor predito pelo modelo preditivo para essa observação. Note que valores mais próximos de 1 da métrica R^2 indicam que as predições são quase perfeitas. Por outro lado, R^2 pode assumir

valores arbitrariamente negativos assinalando predições de baixa acurácia.

4.3.3.2 Otimização de Perparâmetros

A escolha correta dos parâmetros de entrada dos algoritmos de aprendizado de máquina é feita por meio do método *grid-search* associado com a validação cruzada *K-fold*. O *grid-search* consiste em uma busca exaustiva sobre um conjunto de configurações de parâmetros manualmente definidos. Desse modo, é construído um modelo preditivo para cada configuração de parâmetros que é avaliado posteriormente usando a validação cruzada *K-fold*.

Neste caso por exemplo, foi analisado o método de regressão *Gradient boosting machine*. O Quadro 3 traz os parâmetros definidos a serem usados pela técnica *grid-search*.

Quadro 3 – Quadro de valores fornecido à técnica *grid-search*

Método	**Parâmetro**	**Valores**
Gradient Boosting Machine	Taxa de aprendizado	0.01,0.02, 0.05, 0.1
	Altura máxima da árvore	4,6
	Quantidade de modelos base	10,30,50,70,100,1000

Fonte – Elaborado pelo autor

4.4 PROCEDIMENTO DE COMPARAÇÃO DO DESEMPENHO DE MODELOS GERADOS

O objetivo desta fase é de mostrar o processo de comparação entre modelos gerados utilizando métodos de aprendizado de máquina, *AdaBoost* e *Extremely Randomized Trees* que são propostas neste trabalho e modelos gerados modelos gerados utilizando métodos *Regressão Linear* e *Gradient Boosting Machine* que foram utilizados no trabalho de Farias *et al.* (2017), para cada uma das três métricas do desempenho listada na Seção 4.3.1.1. Visto que, além desses dois métodos de regressão que são propostas neste trabalho, foram implementados também neste trabalho os métodos que foram utilizados por Farias *et al.* (2017), da qual é possível ver os resultados na Seção 5.3.

As comparações são feitas no próximo capítulo, Seção 5.3, onde é possível ver a análise e discussão da eficiência entre os métodos *AdaBoost* e *Extremely Randomized Trees* tanto para características lineares, quanto para características quadráticas e os métodos *Regressão Linear* e *Gradient Boosting Machine* no mesmo contexto.

4.5 COMENTÁRIO GERAL DO CAPÍTULO

Neste capítulo foi apresentada uma abordagem para modelagem de desempenho para sistemas de banco de dados NewSQL. A modelagem de desempenho foi baseada em técnicas de aprendizado de máquinas onde, inicialmente, foram executados experimentos para gerar o conjunto de observações de desempenho. Em seguida foram propostos dois modos de manipulação das características para melhorar a acurácia dos modelos gerados: características lineares e características quadráticas. Também foi mostrada o procedimento de comparação do desempenho entre modelos gerados com métodos de aprendizado de máquina que o Farias *et al.* (2017) usou, com os métodos que são propostos neste trabalho. A metodologia proposta neste capítulo é semelhante a de trabalho de Farias *et al.* (2017) no qual esta proposta é baseada.

5 AVALIAÇÃO EXPERIMENTAL

São realizados experimentos para avaliar a efetividade da abordagem de modelagem de desempenho proposta neste trabalho. Além disso, são discutidos acurácia dos modelos gerados com os métodos de aprendizado de máquina que são propostos neste trabalho em relação aos modelos gerados com métodos que foram usando no trabalho de Farias *et al.* (2017). Os objetivos específicos da avaliação experimental são discutidos nas seções seguintes.

5.1 INFRAESTRUTURA

Foi implementado um protótipo da abordagem de modelagem de desempenho usando a linguagem *Python* juntamente com a automatização dos experimentos da Seção 4.3.1.1. Para métodos de aprendizado de máquina foi utilizado as implementações disponíveis na biblioteca Scikit-learn (PEDREGOSA *et al.*, 2011).

Como citado anteriormente, o *benchmark* YCSB versão 0.15.0 foi utilizado nos experimentos com os seguintes parâmetros: 40 *threads* de clientes, 360000 tuplas na base dados, padrão de acesso uniforme e 1 *byte* como tamanho de cada campo da tupla. Os parâmetros *target* (quantidade de requisições submetidas ao sistema por segundo) e <P_{read}, P_{scan}, P_{update}, P_{insert}> (proporção de cada tipo de requisição submetida ao sistema) são variados conforme descritos nos experimentos a seguir. Os parâmetros restantes foram mantidos com os valores padrões. Foi utilizado o banco de dado NuoDB (INC, 2018) versão 3.0. Entre cada experimento a base de dados foi excluída e carregada novamente.

Esse protótipo foi implantado e testado num servidor (Lenovo SR630, 2 x Processador Intel Xeon Silver 4116, 64 GB DDR4 RDIMM, 1x 256 GB SSD, 2x 1 TB SAS e 4x 1 Gbps) do curso de Mestrado Acadêmico em Ciência da Computação da Universidade Estadual do Ceaŕa, onde YCSB foi executado.

5.2 AVALIAÇÃO DA MODELAGEM DE DESEMPENHO

Por meio de construção de modelos de aprendizado de máquina, a abordagem da modelagem de desempenho visa predizer as métricas utilizando como base a carga de trabalho e o tamanho do servidor. Os experimentos específicos dessa abordagem são projetados para atingir os seguintes objetivos:

- Avaliar o poder preditivo dos modelos de desempenho: este objetivo foi realizado por

meio da validação cruzada *k-fold* adotando a métrica R^2;

- Investigar a complexidade subjacente das diferentes métricas de desempenho;
- Determinar em quais métodos será mais interessante adotar as **características lineares (CL)** ou **quadráticas (CQ)** para os modelos de desempenho.

5.2.1 Resultados Experimentais

Os resultados dos métodos são expressos em termos da métrica R^2 em conjunto com a validação cruzada *10-fold*, com o *grid-search* no parâmetros de entrada do método. Os resultados de dois métodos foram comparados: *AdaBoost* (AB) e *Extremely Randomized Trees* (ERT). A métrica de precisão R^2 pode ser contra intuitiva do ponto de vista de um Administrador de Banco de Dados (DBA). Mas, vale ressaltar que valores de R^2 próximos de 1 indicam predições perfeitas e valores menores que 1 (ou, até mesmo, negativos) representam predições de baixa acurácia.

Além da métrica R^2 com *grid-search*, também foi apresentado um modo gráfico de verificação de acurácia dos modelos gerados que exploram a relação do valor real com o valor predito. Para cada caso, foi treinado um modelo com o conjunto de dados completo e foram gerados predições para os valores de saída reais desse mesmo conjunto de dados. Desse modo, em um gráfico de dispersão, foi adicionado um ponto (x, y) para cada amostra do conjunto de dados, sendo x o valor predito pelo modelo e y o valor real da amostra. Assim, em situações onde os pontos estão próximos da reta $y = x$, as predições são mais acurada, e pontos longe da reta $y = x$ indicam predição menos acuradas.

Com isso, pode observar os casos onde é melhor utilizar características lineares ou características quadráticas para melhorar a acurácia dos modelos de desempenho gerados. Para tal, foram comparados ambos os casos com cada uma das seguintes métricas de desempenho:

1. TRS: Tempo de resposta médio por segundo.
2. RPS: Quantidade de requisições executadas pelo sistema por segundo (vazão).
3. VPS: Número de violações por segundo.

A discussão sobre cada métrica citada acima é apresentada nas subseções seguintes.

5.2.1.1 Tempo de Resposta Médio por Segundo (TRS)

A métrica TRS corresponde a média dos tempos de resposta (latência) de todas as requisições executadas com sucesso pelo sistema. A Tabela 2 exibe pontuação R^2 obtida

com a técnica *grid-search* e *10-fold* usando os métodos AB e ERT. Também são apresentados os resultados quando as transformações de características lineares (CL) e de características quadráticas (CQ) são realizadas.

Tabela 2 – Métrica TRS: Pontuação R^2 obtido com *grid-search* e *10-fold* para características lineares e quadráticas com métodos AB e ERT

	TRS	
	AB,	**ERT**
CL	0.91	0.97
CQ	0.92	0.96

Fonte – Elaborado pelo autor

O método AB apresentou bons resultados quando aplicado com características lineares ($R^2 = 0.91$). Ao utilizar características quadráticas, melhorou um ponto percentual em relação características lineares ($R^2 = 0.92$). Por sua vez, o método ERT foi capaz de se ajustar à curva dos dados de modo mais eficaz, por ser baseado em árvores. Também o resultado das características quadráticas ($R^2 = 0.96$) é menos preciso em relação características lineares com um leve diferença ($R^2 = 0.97$).

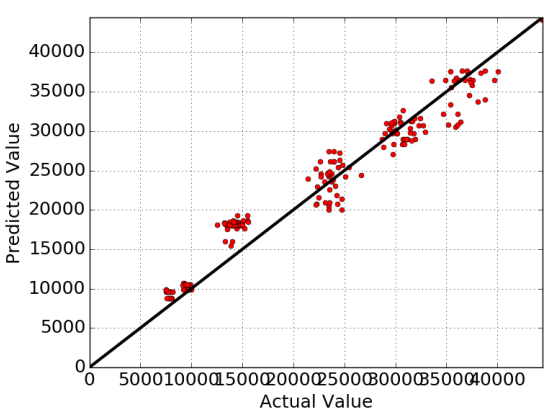
Como indica a pontuação R^2, na Figura 7, os modelos gerados pelo método AB tem os pontos mais distantes da reta $y = x$ e, consequentemente, são modelos de acurácia mais baixa, o que pode ser um erro impraticável para algumas aplicações. No entanto, percebe-se que as amostras situadas no quadrante superior direito obtiveram predições ruins no método AB tanto para características lineares quanto para quadrática.

5.2.1.2 Vazão (RPS)

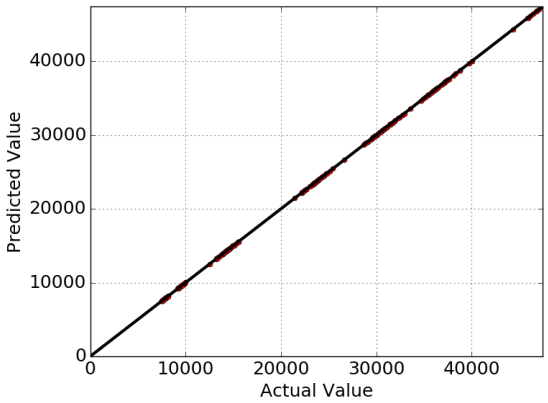
A métrica RPS considera a quantidade de requisições executadas a cada segundo. A abordagem para modelar o desempenho foi aplicada em termos da métrica RPS. A Tabela 3 exibe a pontuação R^2 obtida com a técnica *grid-search* e *10-fold* usando os métodos AB e ERT assim como os resultados quando é feita a transformação de características lineares (CL) e características quadráticas (CQ).

Difrente da métrica TRS, o método AB teve pontuação R^2 menos alta ao se utilizar características lineares ($R^2 = 0.93$) em relação as características quadráticas, com uma pequena diferença ($R^2 = 0.94$) . Já o método ERT, conseguiu predições próximas da perfeição ($R^2 = 0.98$) para ambas as características.

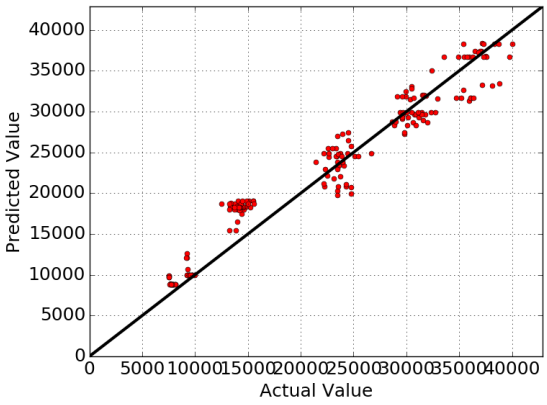
Figura 7 – Métrica TRS: relação entre valor predito e valor real para os modelos gerados



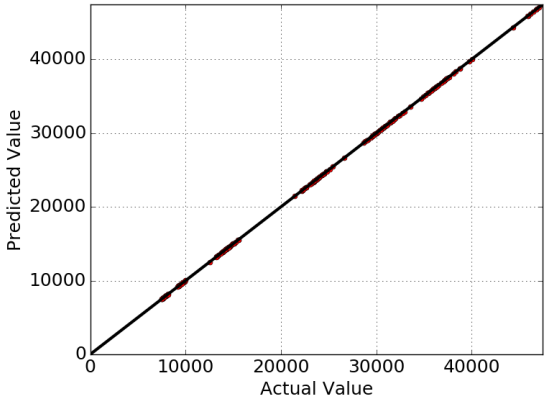
(a) Características lineares - AdaBoost



(b) Características lineares - Extremely Randomized Trees



(c) Características quadráticas - AdaBoost



(d) Características quadráticas - Extremely Randomized Trees

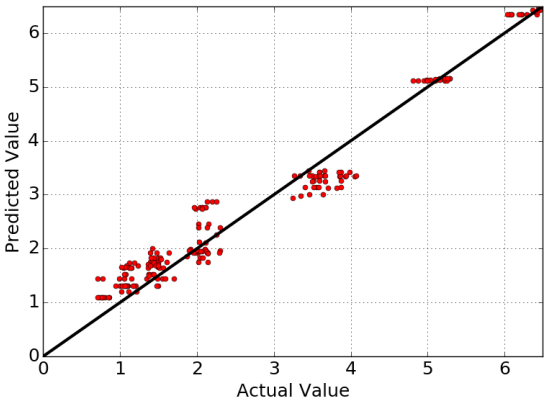
Tabela 3 – Métrica RPS: Pontuação R^2 obtido com *grid-search* e *10-fold* para características lineares e quadráticas com métodos AB e ERT

	RPS	
	AB,	**ERT**
CL	0.93	0.98
CQ	0.94	0.98

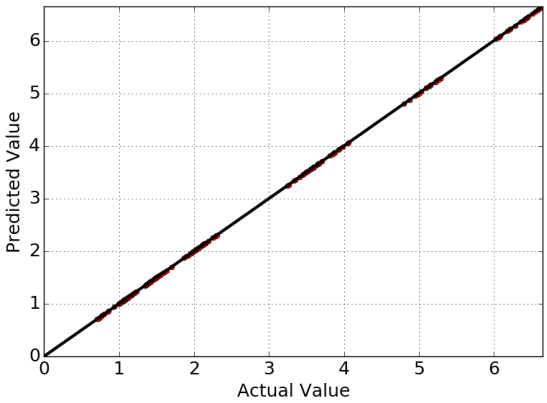
Fonte – Elaborado pelo autor

Similarmente a métrica TRS, como indica a pontuação R^2 , os modelos gerados pelo método AB tem os pontos mais distantes da reta $y = x$ e, diferente da métrica TRS percebe-se que as amostras situadas no quadrante inferior-esquerdo obtiveram predições ruins no método AB tanto para características lineares quanto para quadrática. Por sua vez, os gráficos das Figuras 8b e 8d ilustram as predições com ótimos resultados neste cenário em todas as duas transformações de características.

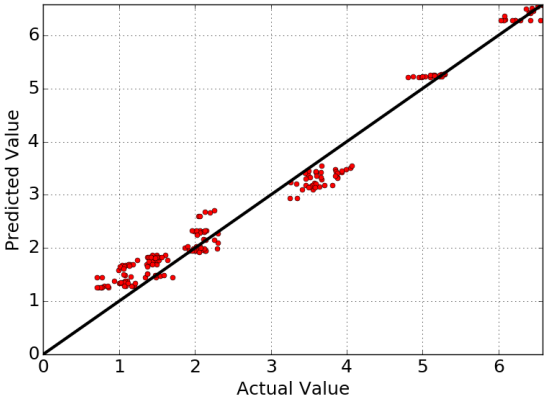
Figura 8 – Métrica RPS: relação entre valor predito e valor real para os modelos gerados



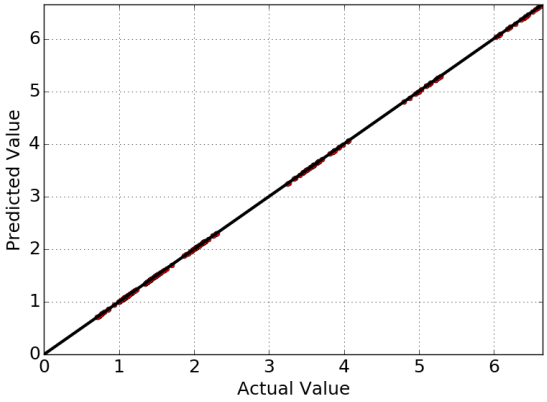
(a) Características lineares - AdaBoost



(b) Características lineares - Extremely Randomized Trees



(c) Características quadráticas - AdaBoost



(d) Características quadráticas - Extremely Randomized Trees

5.2.1.3 Violações por Segundo (VPS)

Similarmente às métricas anteriores, abordagem de modelagem de desempenho foi utilizada com o objetivo de modelar a métrica VPS. Definiu-se o limite de tempo para requisições em 100ms. A Tabela 4 apresenta um comparativo de pontuações R^2 usando AB e ERT com características lineares e quadráticas.

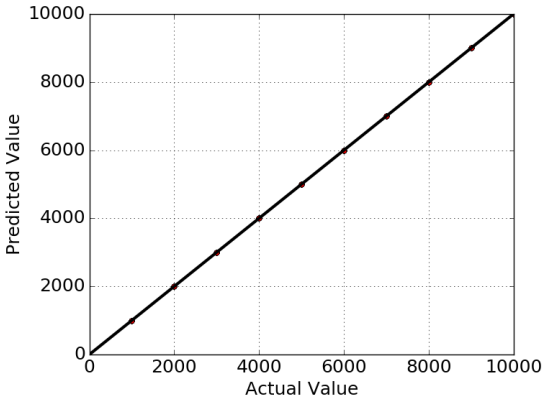
Tabela 4 – Métrica VPS: Pontuação R^2 obtido com *grid-search* e *10-fold* para características lineares e quadráticas com métodos AB e ERT

	VPS	
	AB,	**ERT**
CL	0.98	0.99
CQ	0.99	0.99

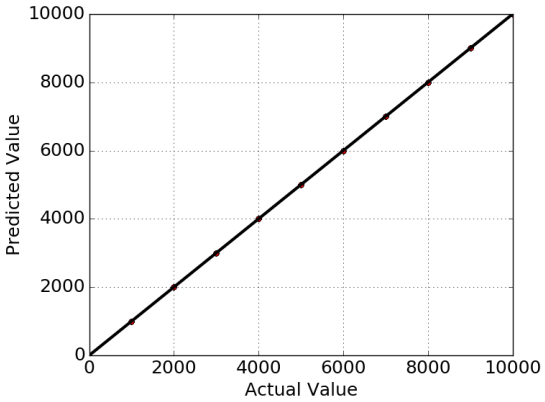
Fonte – Elaborado pelo autor

Como é possível constatar por meio da Tabela 4, a métrica VPS é mais previsível que as anteriores visto que, em todos modelos gerados, a pontuação R^2 foi alta. De qualquer maneira, percebe-se que houve uma melhoria na pontuação R^2 ao usar as característica quadráticas em relação a linear (R^2 = 0.98 para R^2 = 0.99) referente ao método AB. O método ERT obteve acurácia próxima da perfeição em ambos casos (R^2 = 0.99)

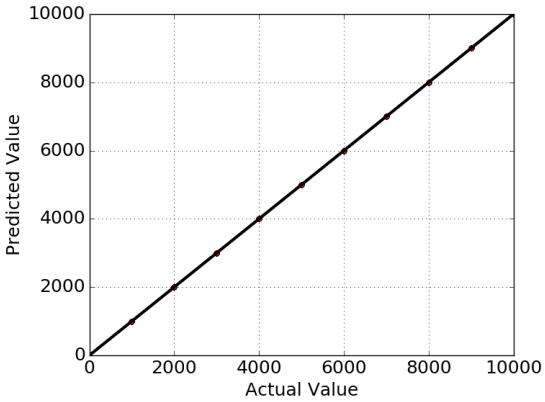
Figura 9 – Métrica VPS: relação entre valor predito e valor real para os modelos gerados



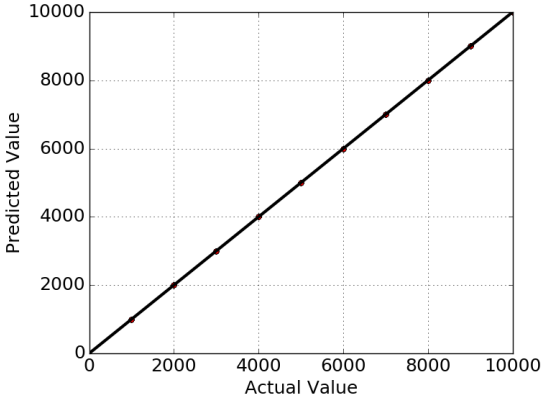
(a) Características lineares - AdaBoost



(b) Características lineares - Extremely Randomized Trees



(c) Características quadráticas - AdaBoost



(d) Características quadráticas - Extremely Randomized Trees

Todos os gráficos da Figura 9 demonstram predições acuradas em todos os métodos muito próximos do valor real. As características linear do método AB tem um leve diferença (R^2 = 0.98) com os demais que apresentam pontuação (R^2 = 0.99).

5.3 AVALIAÇÃO DO DESEMPENHO DOS MODELOS GERADOS

Nesta fase, foi feita a análise comparativa dos modelos gerados pelos métodos de aprendizado de máquina *Regressão Linear* (RL) e *Gradient Boosting Machine* (GBM) que foram

usados no trabalho de Farias *et al.* (2017), com os métodos utilizados neste trabalho, *AdaBoost* (AB) e *Extremely Randomized Trees* (ERT).

5.3.1 Apresentação de Resultados

A Tabela 5 apresenta os resultados de modelos gerados a partir dos métodos de aprendizado de máquinas RL e GBM para características lineares e quadráticas, para as três métricas do desempenho. Nota que os resultados de modelos gerados com métodos RL e GBM para cada uma das três métricas do desempenho não foram discutidos separadamente como na Seção 5.2.1 para os métodos AB e ERT, por serem métodos utilizados no trabalho de Farias *et al.* (2017), entende-se que não seja necessário, eles são apresentados numa única tabela, Tabela 5.

Tabela 5 – Métricas TRS, PRS e VPS: Pontuação R^2 obtido com *grid-search* e *10-fold* para características lineares e quadráticas com métodos RL e GBM

		TRS	RPS	VPS
RL	**CL**	0.74	0.63	0.94
	CQ	0.89	0.82	0.99
GBM	**CL**	0.98	0.98	0.98
	CQ	0.97	0.97	0.99

Fonte – Elaborado pelo autor

A Tabela 6 apresenta os resultados de modelos gerados a partir dos métodos de aprendizado de máquinas AB e ERT que foram ilustrados e discutidos na fase anterior (Seção 5.2.1), para características lineares e quadráticas e para as três métricas do desempenho.

Tabela 6 – Métricas TRS, PRS e VPS: Pontuação R^2 obtido com *grid-search* e *10-fold* para características lineares e quadráticas com métodos AB e ERT

		TRS	RPS	VPS
AB	**CL**	0.91	0.93	0.98
	CQ	0.92	0.94	0.99
ERT	**CL**	0.97	0.98	0.99
	CQ	0.96	0.98	0.99

Fonte – Elaborado pelo autor

5.3.2 Análise e Discussão dos Resultados

Observando os resultados exibidos pelas Tabelas 5 e 6, percebe-se que o método AB apresentou melhores resultados tanto em características lineares quanto em características

quadráticas em relação ao método RL. Os resultados mostram que o método AB se mostrou mais expressivo em características lineares e menos expressivo em características quadráticas, que pode ser observado em todas as métricas do desempenho, exceto em características quadráticas da métrica VPS onde os dois apresentam mesmo resultado ($R^2 = 0.99$), isto porque, apesar de transformações das características aplicada aos métodos para melhorar a qualidade dos modelos preditivos permitir com que RL modele relações não lineares, sua capacidade de capturar superfícies arbitrárias é menos eficiente em relação a AB. Este, apesar de ser um método eficiente e com bons resultados, apresentou resultados menos acurados tanto em características lineares quanto em características quadráticas em relação GBM, exceto na métrica VPS, onde os dois apresentam o mesmo resultado ($R^2 = 0.98$) para características lineares e ($R^2 = 0.99$) características quadrática. Isto se deve ao GBM conseguir capturar naturalmente dependências não-lineares entre as características e as métricas de forma mais eficiente.

De maneira similar a AB, ERT apresentou ótimos resultados, muito mais expressivos tanto em características lineares quanto em características quadráticas em todas as métricas do desempenho em relação a RL, exceto em características quadráticas da métrica VPS, onde os dois apresentam mesmo o resultado ($R^2 = 0.99$). Em média, ERT apresentou resultados similar a GBM, com leve diferença de um ponto percentual em algumas métricas, para mais ou para menos, tanto para ERT quanto para GBM. Os dois métodos apresentaram excelente resultados neste cenário por serem baseadas em árvore, por isso conseguem ajustar à curva dos dados de modo mais eficaz.

5.4 COMENTÁRIO GERAL DO CAPÍTULO

Neste capítulo foram apresentados e discutidos os resultados dos modelos gerados da abordagem de modelagem do desempenho onde foi medido a capacidade de generalização dos modelos de desempenho. De acordo com as análises é possível verificar em qual método é mais vantajoso usar características lineares ou quadráticas, devido ao perfil dos resultados gerados pelos testes neste cenário. O método *AdaBoost* é mais recomendado para usos com características quadráticas. Já o método *Extremely Randomized Trees* se manteve indiferente ao uso de características lineares ou quadráticas, por isso, recomenda-se usar características lineares por ser a abordagem mais simples.

Além da abordagem de modelagem de desempenho, também foram apresentados e comparados resultados dos modelos gerados com os métodos de aprendizado de máquina

Regressão Linear e *Gradient Boosting Machine* que foram usados no trabalho de Farias *et al.* (2017), com os resultados dos modelos gerados com métodos *AdaBoost* e *Extremely Randomized Trees* propostos por este trabalho. Conclui-se que é mais vantajoso usar o método *AdaBoost* em relação a *Regressão Linear* devido os bons resultados apresentado neste cenário. E devido a similaridade dos resultados apresentados usando métodos *Extremely Randomized Trees* e *Gradient Boosting Machine*, recomenda-se usar qualquer um dos dois métodos. Por apresentarem ótimos resultados neste cenário, conclui-se que são mais eficientes em relação a *Regressão Linear* e *AdaBoost*.

6 CONCLUSÕES E TRABALHOS FUTUROS

6.1 CONCLUSÕES

Esse trabalho apresentou uma nova abordagem para modelagem do desempenho para sistemas de bancos de dados NewSQL. A abordagem proposta gera modelos que são capazes de predizer métricas de desempenho baseadas no SLA, capturando efeitos não-lineares causados por aspectos de distribuição e de concorrência. Essa estratégia gera modelos preditivos para estimar métricas de desempenho por meio de técnicas de aprendizado de máquina. O conjunto de treinamento para treinar os modelos de aprendizado foi gerado a partir de uma extensa experimentação do SGBD alvo, sobre o qual foram testados várias configurações de carga de trabalho. Dos experimentos foram extraídos apenas características que são genéricas a qualquer SGBD NewSQL: tamanho do servidor e parâmetros sobre a carga de trabalho (t, t_{read}, t_{scan}, t_{insert}, t_{update}).

Foram investigados aspectos de como as configurações do servidor e da carga de trabalho influenciam de modo não-linear no desempenho. Foi proposta uma transformação de características (lineares e quadráticas) para melhorar a precisão dos métodos no tratamento dos aspectos que contribuem para a alteração do desempenho (Seção 4.3). Técnicas de aprendizado baseadas em árvore, como o *Extremely Randomized Trees* e o *Gardien Boosting Machine*, conseguem capturar naturalmente dependências não-lineares entre as características e as métricas.

Ainda foram comparados os modelos gerados com diferentes métodos, os resultados experimentais confirmam que todos os métodos conseguem gerar modelos acurados, sendo que em ordem crescente, os modelos lineares apresentam menor acurácia, depois *AdaBoost* e por último *Gardien Boosting Machine* e *Extremely Randomized Trees*.

6.2 TRABALHOS FUTUROS

Existem algumas oportunidades de trabalhos futuros que derivam do presente trabalho, das quais se destacam: (i) testar esta abordagem de modelagem de desempenho com mais de um servidor , isto é, um *cluster* com mais de um nó; (ii) aplicar modelos de elasticidade, isto é, provisionamento elástico automático baseado em modelos de desempenho dessa abordagem, que adiciona e remove recursos em tempo de execução de acordo com as características da carga de trabalho corrente; (iii) testar cargas de trabalho reais de sistemas de bancos de dados NewSQL; e (iv) modelagem de desempenho *online* para sistemas de bancos de dados NewSQL.

REFERÊNCIAS

ALPAYDIN, E. **Introduction to machine learning**. [S.l.]: MIT press, 2014.

DATE, C. J. **Introdução a sistemas de bancos de dados**. [S.l.]: Elsevier Brasil, 2004.

DIANA, M. D.; GEROSA, M. A. Nosql na web 2.0: Um estudo comparativo de bancos não-relacionais para armazenamento de dados na web 2.0. In: **IX Workshop de Teses e Dissertações em Banco de Dados**. [S.l.: s.n.], 2010. v. 9.

ELMASRI, R.; NAVATHE, S. B.; MORAIS, R. de O. Sistemas de banco de dados. Pearson Addison Wesley, 2005.

FARIAS, V. A.; SOUSA, F. R.; MAIA, J. G. R.; GOMES, J. P. P.; MACHADO, J. C. Regression based performance modeling and provisioning for nosql cloud databases. **Future Generation Computer Systems**, Elsevier, 2017.

GRAY, J.; HELLAND, P.; O'NEIL, P.; SHASHA, D. The dangers of replication and a solution. **ACM SIGMOD Record**, ACM, v. 25, n. 2, p. 173–182, 1996.

HALL, M.; FRANK, E.; HOLMES, G.; PFAHRINGER, B.; REUTEMANN, P.; WITTEN, I. H. The weka data mining software: an update. **ACM SIGKDD explorations newsletter**, ACM, v. 11, n. 1, p. 10–18, 2009.

KAUR, K.; SACHDEVA, M. Performance evaluation of newsql databases. In: IEEE. **Inventive Systems and Control (ICISC), 2017 International Conference on**. [S.l.], 2017. p. 1–5.

KHAN, H. U. *et al.* Currencies analysis based on stability: Using apriori-algorithm. **International Journal of Business and Management**, v. 5, n. 5, p. 57, 2010.

LAKSHMANAN, L. V.; SADRI, F.; SUBRAMANIAN, I. On e ciently implementing schemasql on a sql database system. In: VLDB. [S.l.], 1999.

MEMSQL. **Architecture MeMSQL**. 2015. Disponível em: <https://www.memsql.com/content/architecture/>.

MONARD, M. C.; BARANAUSKAS, J. A. Conceitos sobre aprendizado de máquina. **Sistemas inteligentes-Fundamentos e aplicações**, v. 1, n. 1, p. 32, 2003.

MONIRUZZAMAN, A. Newsql: towards next-generation scalable rdbms for online transaction processing (oltp) for big data management. **arXiv preprint arXiv:1411.7343**, 2014.

MONIRUZZAMAN, A.; HOSSAIN, S. A. Nosql database: New era of databases for big data analytics-classification, characteristics and comparison. **arXiv preprint arXiv:1307.0191**, 2013.

MOZAFARI, B.; CURINO, C.; JINDAL, A.; MADDEN, S. Performance and resource modeling in highly-concurrent oltp workloads. In: ACM. **Proceedings of the 2013 acm sigmod international conference on management of data**. [S.l.], 2013. p. 301–312.

NAIK, S. Tier: Table index evaluator and recommender—a proposed model to improve transaction performance in distributed heterogeneous database. In: IEEE. **Soft Computing and its Engineering Applications (icSoftComp), 2017 International Conference on**. [S.l.], 2017. p. 1–8.

NIKRAVESH, A. Y.; AJILA, S. A.; LUNG, C.-H. Cloud resource auto-scaling system based on hidden markov model (hmm). In: IEEE. **Semantic Computing (ICSC), 2014 IEEE International Conference on**. [S.l.], 2014. p. 124–127.

NOWOSIELSKI, A.; KOWALSKI, P. A.; KULCZYCKI, P. A database performance polynomial multiple regression model. In: IEEE. **Computer Science and Information Systems (FedCSIS), 2017 Federated Conference on**. [S.l.], 2017. p. 743–474.

PAVLO, A.; ASLETT, M. What's really new with newsql? **ACM Sigmod Record**, ACM, v. 45, n. 2, p. 45–55, 2016.

PEDREGOSA, F.; VAROQUAUX, G.; GRAMFORT, A.; MICHEL, V.; THIRION, B.; GRISEL, O.; BLONDEL, M.; PRETTENHOFER, P.; WEISS, R.; DUBOURG, V. *et al.* Scikit-learn: Machine learning in python. **Journal of Machine Learning Research**, v. 12, n. Oct, p. 2825–2830, 2011.

PRATI, R. C.; BARANAUSKAS, J. A.; MONARD, M. C. Padronizaçao da sintaxe e informaçoes sobre regras induzidas a partir de algoritmos de aprendizado de máquina simbólico. **Revista Eletrônica de Iniciaçao Cientıfica**, v. 2, n. 3, p. 21, 2002.

RAMAKRISHNAN, R.; GEHRKE, J. **Sistemas de gerenciamento de banco de dados-3**. [S.l.]: AMGH Editora, 2008.

SILA, J. **Primerjava NewSQL podatkovnih baz NuoDB in VoltDB**. Tese (Doutorado) — Univerza v Ljubljani, 2015.

TUDORICA, B. G.; BUCUR, C. A comparison between several nosql databases with comments and notes. In: IEEE. **Roedunet International Conference (RoEduNet), 2011 10th**. [S.l.], 2011. p. 1–5.

YCSB. **Benchmark YCSB**. 2015. Disponível em: <https://github.com/brianfrankcooper/YCSB>.

YUAN, L.-Y.; WU, L.; YOU, J.-H.; CHI, Y. A demonstration of rubato db: A highly scalable newsql database system for oltp and big data applications. In: ACM. **Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data**. [S.l.], 2015. p. 907–912.