Um estudo comparativo sobre algoritmos de mineração de dados para execução na WEB

Ronaldo Canofre M. dos Santos, Eduardo N. Borges, Karina dos Santos Machado

Centro de Ciências Computacionais – Universidade Federal do Rio Grande (FURG)
Caixa Postal 474, 96203-900, Rio Grande – RS

canofre@inf.ufsm.br, eduardoborges@furg.br, karinaecomp@gmail.com

Abstract. *The human ability to analyze and obtain information from existing data has not followed the increase in the stored volume. Thus, knowledge discovery in databases techniques and tools have been discussed and used for several years, during which has also increased the use of Web tools. This paper presents a comparative study of data mining algorithms implemented to run on the Web, with PHP.*

Resumo. *A capacidade humana de analisar e obter informações a partir dos dados existentes não tem acompanhado o crescimento no volume armazenado. Com isso, a utilização de técnicas e ferramentas de descoberta de conhecimento em banco de dados vem sendo discutidas e utilizadas há alguns anos, período no qual também tem crescido o uso de ferramentas web. Este trabalho apresenta um estudo comparativo sobre algoritmos de mineração de dados implementados para execução na Web, com a linguagem PHP.*

1. Introdução

O vasto volume de dados gerados até os dias de hoje já ultrapassa a marca de 4,4 ZB (zettabytes) segundo pesquisa da EMC divulgada em 2014 e é acrescido diariamente tanto pela interação homem/máquina como por equipamentos que não necessitam da intervenção humana. Nesta mesma pesquisa é estimado ainda que em 2020 já tenham sido gerados cerca de 44 ZB ou 44 trilhões de gigabytes [EMC Corporation 2014].

Segundo Han et al. (2011), atualmente não existe mais a era da informação, mas sim a era dos dados. Volumes gigantescos são gerados e armazenados diariamente por inúmeras áreas, tais como ciência, medicina, comércio e engenharia. Transações financeiras, vendas *online*, pesquisas e experimentos realizados, registros médicos e sistemas de monitoramento são alguns exemplos de como eles são gerados.

No entanto, essa grande quantidade de dados passa a ter valor a partir do momento em que se torna possível a extração de conhecimento a partir deles, o que é realizado pelo processo de Descoberta e Conhecimento em Banco de Dados ou KDD (*Knowledge Discovery in Databases*). Neste processo, a Mineração de Dados (MD) é a principal etapa e consiste na combinação de métodos tradicionais de análise de dados com algoritmos sofisticados para processamento de grandes volumes [Tan et al. 2006].

Para realização do processo de KDD, em especial a etapa de mineração de dados, existem inúmeras ferramentas disponíveis, tanto livres como comerciais. Em geral, essas ferramentas necessitam ser instaladas em um equipamento para que executem localmente. Por exemplo: Weka, R, ODM, MDR, KMINE, Pimiento, dentre outras [Camilo and da Silva 2009].

Este trabalho tem como objetivo realizar uma revisão e um estudo comparativo sobre algoritmos de mineração de dados que executem diretamente em um ambiente *Web*, sem a necessidade de instalação de programas, bibliotecas e pacotes. Mais especificamente, revisando ferramentas desenvolvidas na linguagem PHP[1].

Dessa forma, a principal contribuição deste trabalho é a busca e avaliação de ferramentas/algoritmos que possam ser utilizados sem a necessidade de instalação e independentes de Sistema Operacional (SO), facilitando assim a portabilidade e o acesso. Soma-se a estes benefícios a possibilidade de permitir e facilitar o aprendizado prático da Mineração de Dados agilizando, por exemplo, a sua utilização em sala de aula.

O restante deste texto está organizado como segue. A fundamentação teórica sobre KDD e Mineração de Dados está descrita na seção 2. A seção 3 apresenta uma abordagem geral sobre ferramentas de mineração de dados, a metodologia e as bases de dados utilizadas. Na seção 4 são apresentados os resultados, incluindo as avaliações realizadas e, por fim, na seção 5 são apresentadas as conclusões e propostas para trabalhos futuros.

2. Revisão bibliográfica

2.1. *Knowledge Discovery in Databases*

A possibilidade da aplicação de processos de KDD em todos os tipos de dados fortalece a sua utilização para inúmeras aplicações, tais como: gerenciamento de negócios, controle de produção, análise de mercado, pesquisas científicas, dentre outros [Han et al. 2011].

Segundo Fayyad et al. (1996), KDD consiste em um processo não trivial de identificação de novos padrões válidos, potencialmente úteis e compreensíveis, aplicado sobre um conjunto de dados, visando melhorar o entendimento de um problema ou auxiliar na tomada de decisão (Figura 1). Também pode ser classificado também como um processo interativo, iterativo, cognitivo e exploratório, englobando vários passos e tomada de decisões por parte dos analistas envolvidos.

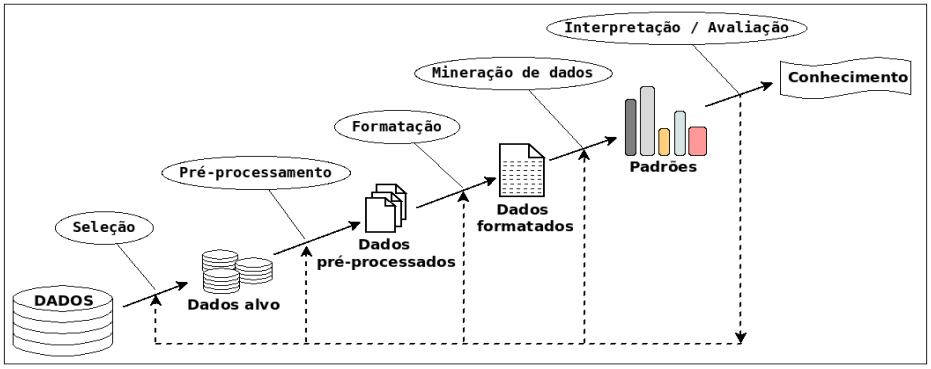

Figura 1. Processo de KDD adaptado de *Fayyad et al. (1996)*.

De acordo com Han et al. (2011), esse processo é dividido nas fases de limpeza, integração, seleção, transformação, mineração, avaliação e apresentação do conhecimento, as quais são divididas em etapas de pré e pós-processamento.

1 Linguagem interpretada utilizada para desenvolvimento de páginas *Web*.

O **pré-processamento** é a etapa mais demorada e trabalhosa do processo de KDD e consiste na preparação dos dados disponíveis na sua forma original a fim de se tornarem apropriados para análise. No entanto, mesmo sendo trabalhosa, é uma etapa importante, pois possibilita a obtenção de melhores resultados quanto ao tempo, custo e qualidade [Tan et al. 2006].

Após a mineração dos dados (abordada na seção 2.2), ocorre a etapa de **pós-processamento**, onde ocorre a interpretação dos padrões minerados e a implantação do conhecimento, assegurando que apenas resultados válidos e úteis sejam incorporados a ferramentas de apoio a decisão.

2.2. Mineração de Dados

Segundo Han et al. (2011), Fayyad et al. (1996) e Tan et al.(2006), a mineração de dados consiste em uma etapa no processo de KKD que busca descobrir ou obter padrões a partir de um grande volume de dados, os quais representem informações úteis. Sua realização consiste na aplicação de algoritmos que se utilizam de técnicas de estatísticas e inteligência artificial, para realizar a classificação de elementos de um conjunto de dados e prever valores de variáveis aleatórias.

É importante ressaltar a diferença entre uma tarefa e uma técnica de mineração. As regularidades ou categorias de padrões que se deseja encontrar com a busca realizada, são especificadas nas tarefas, ao passo que as técnicas de mineração especificam os métodos que possibilitam descobrir os padrões desejados [Bueno and Viana 2012].

As tarefas realizadas pela mineração de dados, podem ser divididas entre preditivas, ou supervisionadas, e descritivas, ou não supervisionadas [Tan et al 2006; Mamon e Rokach 2010]. As tarefas descritivas, como associação, agrupamento e sumarização, realizam a busca de padrões com base a correlação entre os dados. Já as tarefas preditivas, como classificação e regressão, tem o objetivo de prever o valor de um determinado atributo baseado no valor de outros.

Embora essas técnicas ou métodos sejam classificados de acordo com a tarefa realizada, essa separação segue uma linha tênue, visto que alguns métodos preditivos podem realizar tarefas descritivas e vice-versa [Fayyad et al. 1996]:

1. Agrupamento ou *Clustering* – consiste em técnicas de aprendizado não supervisionado que a partir de um conjunto de dados gera subgrupos baseados na semelhança dos objetos [Han et al. 2011];
2. Classificação – processo realizado em duas etapas, uma de aprendizado ou treinamento, onde o modelo de classificação é construído a partir de um conjunto de registros com rótulos conhecidos e a etapa de classificação onde o modelo é utilizado para prever os rótulos de determinado conjunto de dados [Han et al. 2011].
3. Associação – consiste em identificar relacionamentos interessantes escondidos em grandes volumes de dados, os quais podem ser representados na forma de regras de associação ou conjuntos de itens frequentes, podendo ser expressas no formato SE *condição* ENTÃO *resultado* [Tan et al. 2006].
4. Regressão – técnica de modelagem preditiva cuja variável alvo é um valor contínuo[2], principal característica que a diferencia da técnica de

2 Valores contíguos se referem a uma infinita possibilidade de representações dentro de um intervalo.

classificação. Pode ser utilizada, por exemplo, para prever o índice da bolsa de valores com base em outros indicadores econômicos ou a idade de um fóssil baseado na quantidade de carbono-14 presente [Tan et al. 2006].

3. Ferramentas e algoritmos para mineração de dados

Algumas aplicações executáveis utilizadas para mineração de dados, como Knime, Orange Canvas, Rapidminer Studio e Weka, implementam mais de um método e/ou algoritmo para realização das tarefas de mineração, apresentando ainda uma interface gráfica para sua utilização [Boscarioli et al. 2014].

Já outras aplicações podem implementar somente um método e/ou algoritmo, o que em geral pode ser observado em implementações para fins específicos tais como comparações de desempenho ou propostas de novos algoritmos. A revisão realizada neste trabalho a cerca de ferramentas *online* para mineração de dados focou em implementações de ferramentas/algoritmos na linguagem PHP, para uso diretamente no navegador em qualquer sistema operacional, sem necessidade de instalação.

O estudo comparativo e análise das implementações de algoritmos de mineração de dados em PHP foi realizado utilizando bases de dados apropriadas para cada técnica de MD, obtidas do *UCI Machine Learning Repository* [Linchman 2013]. Tais bases de dados são utilizadas como entrada nas respectivas aplicações e na ferramenta Weka, sendo realizando posteriormente a avaliação e comparação dos resultados.

Para avaliação das implementações do algoritmo K-means, foram utilizadas as bases de dados *seeds*, *Wholesale customers* e *Turkiye Student Evaluation.* As implementações do algoritmo *K-Nearest Neighbor (*K-nn), utilizaram as bases *blood transfusion*, *data banknote authentication* e *iris.* Tais bases apresentam variação entre área de aplicação, número de instâncias e total de atributos, sendo abordada na seção seguinte os resultados obtidos por cada conjunto de algoritmos e as bases selecionadas.

4. Resultados obtidos

Durante a revisão foram encontradas diferentes implementações de algoritmos de agrupamento, principalmente o algoritmo K-means, dos quais três implementações (intituladas neste texto como Kmeans01, Kmeans02 e Kmeans03) foram selecionadas para avaliação. Em relação à técnica de classificação, poucos algoritmos foram encontrados, por esse motivo, este trabalho focou a avaliação em implementações do algoritmo *K-Nearest Neighbor* (K-nn) intituladas Knn01, Knn02 e Knn03.

Nos algoritmos encontrados durante a revisão bibliográfica realizada, os dados de entrada são informados diretamente no código, através de arquivos CSV[3] e/ou vetores. Ou seja, essas implementações analisadas, não apresentam interface gráfica. As saídas desses algoritmos são apresentadas no navegador com poucas informações e sem tratamento visual, dificultando a análise dos resultados. As configurações dos parâmetros são também realizadas mediante edição de variáveis e métodos das classes, sendo adotadas em sua maioria as licenças GPL, LGPL e MIT.

3 CSV - *Comma Separated Values*, formato de arquivo que armazena dados tabelados separados por vírgulas

4.1. Implementações do algoritmo K-means

O algoritmo de agrupamento K-means utiliza um método de particionamento baseado em protótipos que busca encontrar um determinado número de grupos (*k*), definidos pelo usuário [Tan et al. 2006]. Os grupos são definidos em torno de um centroide, o qual é a média de um grupo de pontos e em geral não consiste em um valor da base de dados.

Após os grupos formados, um novo centroide é definido com base nos dados deste novo grupo e os grupos são redistribuídos. O algoritmo é finalizado quando os pontos permanecerem no mesmo grupo ou os centroides não sofrerem alteração. Foram estudadas três implementações do algoritmo K-means, definidas como Kmeans01 [Delespierre 2014], Kmeans02 [Yokoyama 2011] e Kmeans03 [Roob 2014].

A saída de cada implementação estudada foi modificada de forma a padronizar o resultado em todas as implementações, com a finalidade de facilitar o entendimento e as avaliações, exibindo assim os centroides de cada *cluster,* identificados pelo seu índice numérico, seguido da quantidade de objetos do grupo no seguinte formato: *Cluster Y [* $x_1,x_2,x_3,...,x_n$*]: N points*.

As avaliações foram realizadas considerando o número de grupos, número de instâncias em cada grupo e como os resultados são exibidos. Como os algoritmos avaliados não implementam nenhuma métrica de validação de agrupamento (como por exemplo DBI, Sillhuette, C-index, etc.) [Tomasini et al. 2016] , as mesmas não foram utilizadas para avaliar a qualidade dos resultados obtidos com cada algoritmo.

Com relação ao número de grupos, as implementações K-means encontradas permitem a definição de $k > n$, não respeitando a premissa de que a quantidade de *clusters* definidas deve ser menor ou igual a quantidade de objetos existentes ($k \leq n$), permitindo assim a ocorrência de grupos vazios. A aplicação Kmeans02 trata estes grupos, transformando o elemento mais distante de qualquer um dos centroides em um *cluster* e, neste caso, a quantidade de *clusters* gerados é inferior ao definido em *k*.

A inicialização dos centroides é realizada de forma randômica nas implementações, sendo possibilitado pela aplicação Kmeans01 uma variação denominada K-means++ que permite uma inicialização alternativa do agrupamento [Arthur and Vassilvitskii 2007] e pela kmeans02, limitado entre o valor mínimo e máximo presente na base de dados. A condição de parada adotada consiste na estabilidade dos centroides, ou seja, quando os mesmos não sofrem mais alteração, sendo tal informação obtida através da exibição dos passos do algoritmo.

Com relação aos atributos, ocorre apenas a restrição ao número de dimensões da base de dados, pela implementação Kmeans03. A implementação Kmeans01 [Delespierre 2014] permite um número ilimitado de atributos, segundo sua documentação, sendo a maior base utilizada, composta por 33 atributos. As alterações no valor de *k* refletem no total de agrupamentos vazios, na igualdade da distribuição das instâncias.

A implementação Kmeans02 [Yokoyama 2011] necessitou que os limites de tempo e uso de memória do servidor fossem alterados, quando utilizada a base *Turkiye Student Evaluation*, sendo tais alterações realizadas através das funções *set_time_limit* e *init_set* [PHP Documentation Group 2015], as quais respectivamente alteram o tempo limite de execução e o valor para quantidade de memória a ser utilizada pelo servidor.

Por fim, a implementação Kmeans03 [Roob 2014] suporta somente agrupamentos formados por duas dimensões, sendo por este motivo avaliada com a base de dados disponibilizada no próprio algoritmo, o qual é composto por 19 coordenadas de um plano cartesiano, entre as coordenadas (0,0) e (20,20). Dessa forma, considerando a quantidade reduzida de objetos possível de ser analisada, foram também utilizadas poucas variações de valores para *k,* sendo observado o surgimento mais constante de *clusters* vazios para os valores mais elevados. Devido a essa restrição, essa implementação não foi executada com as bases de dados do UCI.

4.1.1. Comparação Dos Resultados Das Implementações K-means

Para os resultados das implementações Kmeans01 e Kmeans02, as quais permitiam alterar a base de dados a ser analisada, foram utilizadas além das mesmas bases de dados, um valor único para *k*. Buscando complementar a avaliação realizada, as bases de dados foram também utilizadas na implementação do K-means na ferramenta Weka.

A Tabela 1 apresenta o resultado o total de instâncias em cada grupo, referente a execução de cada algoritmo e do aplicativo Weka, para cada uma das bases de dados, com *k=5*, sendo as bases de dados identificadas como segue: 01-*seeds,* 02-*Wholesale customers* e 03-*Turkiye Student Evaluation*.

Tabela 1. Resultados das implementações Kmeans01, Kmeans02 e Weka.

Base de dados	Instâncias/ Atributos	Instâncias por Implementação		
		Kmeans01	Kmeans02	Weka
01	210/7	[42,64,16,46,42]	[49, 49, 55, 42, 15]	[14, 46, 50, 48, 52]
02	440/8	[113,63,23,6,235]	[235, 113, 63, 6, 23]	[239, 98, 8, 36, 59]
03	5820/33	[1342, 901, 1140, 1261, 1176]	[1344,1175, 902, 1140, 1259]	[760, 731, 1971, 1622, 736]

Devido as aplicações analisadas não implementarem métricas de validação, e a inicialização dos centroides ser realizada de forma aleatória, gerando *clusters* distintos a cada execução, os resultados apresentados foram obtidos a partir de uma sequência de 10 execuções, sendo selecionados os mais recorrentes. Dessa forma, é possível avaliar que os resultados das implementações Kmeans01 e Kmeans02 apresentam uma mesma quantidade de instâncias ou quantidades idênticas, tais como os [...42,] e [...1140,] para as bases 01 e 03 respectivamente e uma mesma distribuição para base 02.

Quando comparados com os resultados do Weka, também foram encontrados *clusters* com o mesmo número de instâncias, como o valor [..46,] presente nos resultados da base 01 para o Kmeans01 e o Weka, além de valores muito próximos. Cabe ressaltar que as variações de resultados observadas durante as execuções das aplicações Kmeans01 e Kmeans02 não foram percebidas durante a execução do Weka.

4.2. Implementações do algoritmo k-NN

O método de classificação baseado no vizinho mais próximo utiliza a técnica de descoberta baseada em instância, a qual requer uma medida de proximidade para classificar os objetos, sendo geralmente utilizado no reconhecimento de padrões [Lima 2011]. Como os objetos são representados como pontos por um classificador, a forma mais comum de determinar as proximidades é através do cálculo da distância entre os pontos. Neste trabalho foram analisadas três implementações do algoritmo k-NN:

Knn01 [Degerli 2012], Knn02 [Mafrur 2012] e Knn03 [Houweling 2012].

Essas implementações apresentam similaridades entre suas características, destacando-se a utilização o método Euclidiano adotado para obtenção das distâncias e a definição da instância a ser classificada, que não apresenta restrições fixas com relação a novos valores. As demais características individuais são abordadas a seguir.

A implementação denominada como Knn01 [Degerli 2012] é a mais limitada dentre as analisadas, restringindo o número de atributos a 2 e fixando a definição de vizinhos mais próximos (*k=4*) e as classes, diretamente no código fonte. Dessa forma, as avaliações para esta aplicação se baseiam somente na base de dados disponibilizada As informações de distâncias e de vizinhos mais próximos, somente são visualizadas através da impressão de um vetor pré-formatado, não permitindo a recuperação dos dados para uma exibição mais intuitiva.

A implementação Knn02 [Mafrur 2012], restringe a 4 o número de atributos da base de dados, não incluindo a classe a qual o objeto pertence, permitindo a definição do número de vizinhos (*k*) a ser utilizada. A classificação das instâncias é realizada através da posição da mesma na base de dados, sendo assim necessário incluir uma nova instância na base para que a mesma possa ser classificada. Esta implementação necessitou ainda que os limites de tempo de execução do servidor fosse alterados de seu valor padrão, quando utilizada a base *banknote authentication.* Tais alterações foram realizadas através do uso da função *set_time_limit* [PHP Documentation Group 2015], definido um novo tempo limite de execução.

A última implementação do algoritmo k-NN analisada, Knn03 [Houweling 2012], não apresenta restrições quando a quantidade de atributos, permitindo a definição de um peso para cada atributo, conforme proposto por Paredes e Vital (2006). A classificação realizada por esta aplicação, ocorre de forma distinta com relação as demais implementações k-NN, não definindo o número de vizinhos mais próximo a serem analisado (*k*), utilizando para tal, a média das distâncias de cada classe para definir a classe de um novo objeto.

4.2.1. Comparação Dos Resultados Das Implementações K-NN

Visando realizar uma comparação entre os resultados das implementações Knn02 e Knn03, os quais possibilitam a execução com uma mesma bases de dados, foram realizados testes com um mesmo conjunto de instâncias a serem classificadas, sendo definidos para um valor de *k=5* e o peso dos atributos de Knn03 foi mantido com o valor 1, com o objetivo de não influenciar nos resultados.

Ainda, buscando complementar a avaliação realizada, foram também analisadas através da aplicação Weka, utilizando as mesmas definições adotadas nas aplicações analisadas. Para realização dos testes foram utilizadas instâncias já classificadas, existentes nas bases de dados, permitindo assim verificar a acurácia das classificações realizadas. Assim sendo, os resultados obtidos demonstraram um comportamento similar entre os resultados das aplicações, onde as mesmas realizaram tanto classificações corretas como incorretas para todas as instâncias analisadas.

Conforme podemos observar na Tabela 2 do conjunto de teste utilizado, a implementação Knn03 apresentou uma média de acerto melhor, sendo para esse pequeno conjunto de teste mais efetivo do que a implementação do Weka.

Tabela 2. Resultados das implementações Knn02 e Knn03 e Weka.

Base de Dados	Instância/ Atributo	Atributos da instância de teste	Classe Correta	Knn02	Knn03	Weka
01	748/4	(1,24,6000,77)	Não	Não	Não	Sim
		(4,6,1500,22)	Sim	Não	Sim	Sim
02	1372/4	(-0.4928,3.060, -1.8356,-2834)	Verdadeira	Verdadeira	Verdadeira	Verdadeira
		(0.6636,-0.0455, -0.1879,0.2345)	Verdadeira	Verdadeira	Falsa	Verdadeira
03	150/4	(4.9,2.0,4.0,1.7)	Virginica	Versicolor	Virginica	Versicolor
		(5.9,3.2,4.8,1.8)	Versicolor	Virginica	Virginica	Virginica
Acurácia média				50%	66%	50%

5. Conclusão e trabalhos futuros

A utilização de aplicações *online* tem almejado facilitar o cotidiano dos usuários, abstraindo questões presentes em aplicações *desktop*[4] e também favorecendo a portabilidade entre os sistemas operacionais, dentre outros fatores. Dessa forma, a utilização de ferramentas *Web* para descoberta de conhecimento possibilita a realização destas tarefas, independente de instalação e sistema operacional, favorecendo o uso em ambientes de aprendizagem por exemplo.

A análise inicial das implementações encontradas levou em consideração os dados de entrada, a apresentação da saída, a forma de uso da aplicação e a licença utilizada, sendo selecionadas para uma avaliação mais detalhada, as aplicações que possibilitam a modificação dos dados de entrada e a execução através do navegador. Devido à inexistência de interface gráfica, as configurações foram realizadas diretamente no código fonte.

A realização de testes com bases de dados obtidas no repositório UCI, seguindo uma configuração padrão para as execuções e instâncias cujas classes eram previamente conhecidos no caso do k-NN. Os resultados gerados foram comparados também com os obtidos pelo *software* Weka, buscando uma maior confiabilidade nos resultados.

Foi possível observar um comportamento comum entre as saídas de todas as aplicações, as quais apresentaram tanto classificações corretas e incorretas no caso do k-NN e para o K-means, resultados muito próximos e em alguns casos iguais, em relação ao número de instâncias em cada agrupamento. No entanto, as avaliações realizadas possibilitaram alcançar os objetivos iniciais propostos por este trabalho, demonstrando a ausência de ferramentas *web* completas desenvolvidas em PHP que possibilitem uma utilização similar as ferramentas *desktop*, permitindo também enumerar inúmeras aplicações únicas de algoritmos, desenvolvidas em PHP, as quais possibilitam a utilização por meio de navegadores.

Os testes realizados também contribuíram com a comprovação da capacidade de utilização de ferramentas de mineração de dados de forma *online* demonstrando um correto funcionamento e sem custos de processamento, possibilitando ainda a utilização multiplataforma.

4 Ferramentas instaladas no computador.

Assim sendo, a pesquisa realizada demonstra uma vasta área de estudos, proporcionando a realização de trabalhos futuros, tanto na área de desenvolvimento como pesquisa. Algumas implementações possíveis seriam de aplicações individuais, tais como as analisadas, no entanto, de forma mais intuitivas, com interface gráfica e prontas para uso. Também a elaboração de uma ferramenta completa composta por vários algoritmos, tal como as aplicações *desktop* existentes.

Referências

Arthur, D. and Vassilvitskii, S. (2007). k-means++: The advantages of careful seeding. *In Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 1027–1035. Society for Industrial and Applied Mathematics.

Boscarioli, C., Viterbo, J. and Teixeira, M. F. (2014). Avaliação de aspectos de usabilidade em ferramentas pra mineração de dados. *Anais da I Escola Regional de Sistemas de Informação do Rio de Janeiro*, 1(1):107-114.

Bueno, M. F. and Viana, M. R. (2012). Mineração de dados: aplicações, eficiência e usabilidade em ferramentas para mineração de dados. *Anais do congresso de Iniciação Científica do INATEL*, 1(1):86–95.

Camilo, C. O. and da Silva, J. C. (2009). Mineração de Dados: Conceitos, tarefas métodos e ferramentas. Relatório técnico, Instituto de Informática. Universidade Federal de Goiás, Goiânia.

Degerli, O. (2012). Data Mining Algorithms' Application with PHP. Disponível em: https://github.com/onurdegerli/data-mining. Acesso em: setembro de 2015.

Delespierre, B. (2014). PHP K-Means. Disponível em: https://github.com/bdelespierre/php-kmeans. Acesso em: setembro de 2015.

EMC Corporation (2014). The digital universe of opportunities: Rich data and the increasing value of the internet of things. Disponível em: http://brazil.emc.com/leadership/digital-universe/2014iview/executive-summary.htm. Acesso em: agosto de 2015.

Fayyad, U. M., Piatetsky-Shapiro, G., and Smyth, P. (1996). Advances in knowledge discovery and data mining. chapter From Data Mining to Knowledge Discovery: An Overview, pages 1–34. American Association for Artificial Intelligence, Menlo Park, CA, USA.

Han, J., Kamber, M. and Pei, J. (2011). *Data Mining Conceps and Techniques*. The Morgan Kaufmann Series in Data Managemente Systems. Elsevier Science, 3ª edition. São Francisco. USA.

Houweling, F. (2012). Knn prototype php. Disponível em: https://github.com/FrankHouweling/KnnPrototypePhp. Acesso em: setembro de 2015.

Lima, G. F. (2011). Classificação Automática de Batidas de Eletrocardiogramas. Trabalho de graduação, Curso de Ciência da Computação, Instituto de Informática. Universidade Federal do Rio Grande do Sul/RS.

Mafrur, R. (2012). Knn php. Disponível em: https://github.com/rischanlab/knn php. Acesso em: setembro de 2015.

Paredes, R. and Vidal, E. (2006). Learning Weighted Metrics to Minimize Nearest-Neighbor Classification Error, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(7):1100-1110.

PHP Documentation Group (2015). Manual do PHP. Disponível em: http://php.net/manual/pt_BR/. Acesso em: outobro de 2015.

Roob, S. (2014). Php k-means. Disponível em: https://github.com/simonrobb/php-kmeans. Acesso em: setembro de 2015.

Tan, P., Steinbach, M., and Kumar, V. (2006). *Introduction to Data Mining*. Pearson international Edition. Pearson Addison Wesley.

Tomasini, C., Emmendorfer, L., Borges, E. and Machado, K. A methodology for selecting the most suitable cluster. In: ACM/SIGAPP Symposium on Applied Computing, 2016 (to appear).

Yokoyama, S. (2011). K-means php. Disponível em: https://github.com/abarth500/K-Means-PHP. Acesso em: setembro de 2015.