UNIVERSIDADE FEDERAL DO PARÁ
INSTITUTO DE CIÊNCIAS EXATAS E NATURAIS
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Fernando Fabio Dias Gama da Mata

Investigando Métodos Inteligentes para Detecção de Anomalias em Comportamento de Insetos Sociais

Belém – Pará – Brasil
2017

UNIVERSIDADE FEDERAL DO PARÁ
INSTITUTO DE CIÊNCIAS EXATAS E NATURAIS
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Fernando Fabio Dias Gama da Mata

Investigando Métodos Inteligentes para Detecção de Anomalias em Comportamento de Insetos Sociais

Belém – Pará – Brasil
2017

UNIVERSIDADE FEDERAL DO PARÁ
INSTITUTO DE CIÊNCIAS EXATAS E NATURAIS
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Fernando Fabio Dias Gama da Mata

Investigando Métodos Inteligentes para Detecção de Anomalias em Comportamento de Insetos Sociais

Dissertação apresentada à Universidade Federal do Pará, como parte dos requisitos do Programa de Pós-Graduação em Ciência da Computação, para obtenção do grau de Mestre em Ciência da Computação.

Área de Concentração: Sistemas Inteligentes

Orientador: Prof. Dr. Gustavo Pessin

Belém – Pará – Brasil
2017

Dados Internacionais de Catalogação na Publicação (CIP)
Sistema de Bibliotecas da Universidade Federal do Pará
Gerada automaticamente pelo módulo Ficat, mediante os dados fornecidos pelo(a) autor(a)

F118i Fabio Dias Gama da Mata, Fernando
Investigando Métodos Inteligentes para Detecção de Anomalias em Comportamento de Insetos Sociais / Fernando Fabio Dias Gama da Mata. - 2017.
63 f. : il. color.

Dissertação (Mestrado) - Programa de Pós-graduação em Ciência da Computação (PPGCC), Instituto de Ciências Exatas e Naturais, Universidade Federal do Pará, Belém, 2017.
Orientação: Prof. Dr. Gustavo Pessin

1. Aprendizagem de Máquina. 2. Detecção de Anomalia. 3. Janela Deslizante. 4. Polinização. 5. Abelhas. I. Pessin, Gustavo , *orient.* II. Título

CDD 006.3

UNIVERSIDADE FEDERAL DO PARÁ
INSTITUTO DE CIÊNCIAS EXATAS E NATURAIS
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

FERNANDO FABIO DIAS GAMA DA MATA

INVESTIGANDO MÉTODOS INTELIGENTES PARA DETECÇÃO DE ANOMALIAS EM COMPORTAMENTO DE INSETOS SOCIAIS

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal do Pará como requisito para obtenção do título de Mestre em Ciência da Computação, defendida e aprovada em 14/12/2017, pela banca examinadora constituída pelos seguintes membros:

Prof. Dr. Gustavo Pessin
Orientador – PPGCC/UFPA

Prof. Dr. Jefferson Magalhães de Morais
Membro Interno – PPGCC/UFPA

Prof. Dr. José Reginaldo Hughes Carvalho
Membro Externo – UFAM

Prof. Dr. Sandro José Rigo
Membro Externo – UNISINOS

Visto:
Prof. Dr. Bianchi Serique Meiguins
Coordenador do PPGCC/UFPA

Dedico esta dissertação à minha família.

Agradecimentos

Primeiramente a Deus por ter me conduzido nesta jornada estando comigo a todo o momento especialmente nos mais desafiadores.

Ao meu orientador de mestrado Gustavo Pessin pela sua disposição em colaborar desde o começo fornecendo ajuda técnica para a construção deste trabalho e por ter acreditado no meu potencial de crescimento acadêmico e profissional.

Agradeço aos meus pais Sandra e Raimundo, meu padrasto Alberto, minha vó Albélia assim como todos os meus irmãos por estarem sempre presentes na minha vida dando apoio incondicional as minhas decisões bem como ter colaborado em todos os sentidos para a consecução de mais esta etapa da minha vida.

À minha amada esposa Samayra, por estar presente nos bons e maus momentos pela sua paciência, pelo consolo, tendo se dedicado a me prestar ajuda necessária sempre que precisei.

Também a todos meus familiares que direta ou indiretamente colaboraram de acordo com suas possibilidades nessa jornada.

A todos meus amigos do laboratório de robótica do Instituto Tecnológico Vale (ITV) Bruno, Eduardo, Helder, Jair, Hanna pela colaboração com experiências, sugestões e parcerias para a execução deste trabalho.

Ao Programa de Pós Graduação em Ciência da Computação (PPGCC) da Universidade Federal do Pará (UFPA), professores, funcionários por estarem presentes e dispostos a colaborar com a minha formação.

Ao Instituto Tecnológico Vale (ITV) pela colaboração dos profissionais e disponibilização do espaço físico para execução do projeto e realização de reuniões.

À Coordenadoria de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) pelo financiamento da bolsa no qual se mostrou um apoio muito importante para o desenvolvimento deste trabalho.

A todos aqueles não mencionados que contribuíram para a conclusão desta etapa da minha vida.

“A persistência é o menor caminho do êxito”.

(Charles Chaplin)

Resumo

Abelhas são um dos mais importantes polinizadores uma vez que auxiliam na reprodução das plantas assegurando a produção de sementes e frutos. Elas são importantes tanto na polinização quanto na produção de mel o que beneficia pequenos e grandes agricultores. No entanto, nos últimos anos, a população de abelhas vem diminuindo de maneira significativa em escala global. Neste cenário, a compreensão do comportamento das abelhas tornou-se uma questão de grande importância e preocupação na tentativa de encontrar as possíveis causas desta situação. Neste trabalho, nosso objetivo é propor, desenvolver e investigar metodologias que combinem métodos não supervisionados para detecção de anomalia baseado em distância e modelos supervisionados baseados em aprendizagem de máquina. Os resultados mostram que a combinação das técnicas permite a detecção de eventos anômalos em comportamento de insetos de forma satisfatória.

Palavras-chaves: Aprendizagem de máquina; Detecção de Anomalia; Janela Deslizante; Polinização; Abelhas.

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

LOF	Local Outlier Factor
KNN	K-Nearest-Neighbor
MLP	Multilayer Perceptron
SVM	Support Vector Machines
RF	Random Forest
RFID	Radio-Frequency IDentification
OOB	Out-of-bag
WSN	Wireless Sensor Networks

Sumário

1 Introdução

1.1 Contextualização

Abelhas tem um papel muito importante na polinização de espécies vegetais. A polinização consiste na transferência do grão de pólen da antera (órgão masculino) de uma flor para o estigma (órgão feminino) de uma flor. Por meio da polinização, as espécies vegetais trocam gametas e tem frutos em maior quantidade e em melhor qualidade. A transferência de grão de pólen resulta em fertilização e formação de sementes. Flores com maiores taxas de fertilização produzem mais sementes e, como consequência, maiores frutos. O sucesso de uma polinização resulta em frutos de qualidade e com maior simetria. Por outro lado, frutos inadequadamente polinizados se tornam muitas vezes menores e deformados. A presença numerosa de polinizadores durante a floração é de suma importância para produzir uma colheita sustentável. Além disso, abelhas nativas ajudam a diminuir o risco de polinização inadequada (LEE, 2010).

A figura 1 apresenta frutos com diferentes tipos de polinização, sendo os frutos, central e a esquerda com desenvolvimento incompleto devido à autopolinização passiva, enquanto o fruto a direita apresenta desenvolvimento completo devido a polinização adequada. Apesar de existirem outros agentes polinizadores tais como o vento (anemofilia), água, pássaros, morcegos e outros animais, a polinização por insetos (entomofilia) é a mais comum onde a grande maioria é formada por abelhas.

Figura 1 – Frutos com diferentes tipos de polinização: as duas primeiras à esquerda representam frutos com autopolinização passiva enquanto que a última foi polinizada por insetos. **Fonte**: (LEE, 2010)

De modo geral as abelhas são responsáveis pela polinização de culturas na ordem de mais de US$ 19 bilhões e pela produção de cerca US$ 385 milhões em mel por ano somente nos Estados Unidos. Na Austrália, a indústria se beneficia da produção de abelhas com US$

92 milhões (FOTH; BLACKLER; CUNNINGHAM, 2016). O trabalho de (GIANNINI et al., 2015) apresentou um estudo que identificou 75 culturas agrícolas brasileiras. Dentre as 250 espécies de polinizadores 87% são abelhas. Destaque para abelha *Melipona fasciculata* (uruçu cinzenta), utilizada em nosso estudo, indicada pela polinizações de muitas culturas e com alto valor de produção. Além disso, são abelhas sociais, úteis para manejo uma vez que não apresentam um ferrão funcional sendo inclusive comprovadamente adequadas para polinização em culturas agrícolas.

No cenário brasileiro, têm-se notado uma queda significativa de abelhas africanizadas [1], que tem como características serem mais resistentes a certos patógenos e parasitas em relação as abelhas europeias puras. Infelizmente, existem algumas poucas evidências que tentam justificar este desaparecimento. Alguns estudos indicam que esse cenário está relacionado a um conjunto de fatores como o surgimento de novos tipos de parasitas, utilização de pesticidas, cultivo de monoculturas, ondas eletromagnéticas geradas por torre de telefonia celular, vegetação alterada geneticamente e o manejo inadequado de colmeias (MESSAGE; TEIXEIRA W.AND JONG, 2012); (RATNIEKS F.; CARRECK N., 2010).

Este trabalho é parte do projeto de microssensores proposto por (SOUZA P. et al., 2017), onde etiquetas eletrônicas (*Radio-Frequency IDentification – RFID*) são coladas em abelhas a fim de aprimorar o conhecimento sobre o comportamento desses insetos. Conhecer o comportamento das abelhas é importante, uma vez que isto auxilia produtores a entender melhor as atividades das abelhas relacionadas com a polinização e assim tomar medidas que possam elevar a produção de alimentos, melhorar a sua qualidade, bem como traçar estratégias para minimizar eventuais impactos que possam reduzir as atividades das abelhas.

Assim, uma das peças chaves de investigação neste trabalho é buscar uma melhor compreensão sobre os métodos para analisar o comportamento de abelhas. O uso de dados obtidos de sensores RFID não é novidade no meio científico, instituições de pesquisa e universidades têm investido na utilização de etiquetas eletrônicas para análise de dados nos mais variados domínios devido à capacidade de rastreamento com precisão dos objetos e também o seu baixo custo.

O declínio da população das abelhas é uma ameaça para a agricultura. De todo o alimento consumido pela humanidade, estima-se que 35% dependa da ação das abelhas (MESSAGE; TEIXEIRA W.AND JONG, 2012). O processo de polinização ocorre quando abelhas operárias visitam flores e o pólen dessas flores fica retido junto aos pêlos dos insetos. Conforme visitam diferentes flores, promovem a troca de gametas entre as plantas

[1] Abelhas africanizadas são conhecidas como abelhas “assassinas”, foram criadas pelo cruzamento de abelhas africanas com europeias e tem como principal característica a defesa da colmeia. Link: BBC Earth

através do pólen. As operárias de uma única colmeia podem visitar mais de cem mil flores no mesmo dia (HENEIN; LANGWORTHY; ERSKINE, 2009).

Alguns lugares dependem da importação de abelhas para melhorar a produção e o desenvolvimento do fruto (FITZGERALD, 2016). O apicultor pode utilizar sua experiência para se dedicar a criação de rainhas para uso próprio ou comercializando com outros apicultores ou agricultores. Há também aqueles que se dedicam à apicultura para polinização de culturas agrícolas com a utilização de abelhas selvagens ou domesticadas.

Apesar do crescimento global do número de colmeias domesticadas, a quantidade de abelhas tem diminuindo nos Estados Unidos desde a década de quarenta e em muitos países da Europa desde a década de sessenta (POTTS S. et al., 2010). Essa diminuição afeta não somente as plantas oriundas das florestas nativas, mas também tem um impacto significativo na agricultura e, como consequência, na economia.

Em face desta realidade, é de vital importância tentar analisar o comportamento de abelhas rastreando-as, na tentativa de encontrar padrões anômalos que afetam o seu desenvolvimento e seu desempenho como agentes polinizadores. A criação de um sistema que identifique padrões anômalos a partir de dados obtidos das abelhas teria como consequência um enriquecimento de informação e conhecimento necessário para que apicultores ou pequenos agricultores tomem medidas para minimizar os efeitos da polinização incompleta e assim, maximizar a produção de frutos com boa qualidade.

1.2 Detecção de anomalia

Um primeiro conceito importante é o de anomalia ou *outlier*. De acordo com (HAWKINS, 1980) uma anomalia (*outlier*) é uma observação que se desvia muito em relação a outras observações que levam a suspeitas de serem geradas por algum mecanismo diferente. Outra definição aponta que as anomalias são padrões em dados que não estão em conformidade com uma noção bem definida de comportamento normal (CHANDOLA; BANERJEE; KUMAR, 2009). Em seu trabalho (JOHNSON, 1992) define *outlier* como uma observação em um conjunto de dados que seja inconsistente em relação a outras observações do conjunto de dados. Assim, a detecção de anomalia objetiva encontrar padrões em um conjunto de dados que não estão em conformidade com um comportamento esperado (SINGH; UPADHYAYA, 2012).

A figura 2 ilustra a presença de anomalias, à esquerda têm-se a representação de pontos na forma de dados agrupados (*cluster*), à direita têm-se a representação na forma de série temporal. Note também que na figura à esquerda as anomalias estão distantes da distribuição, entretanto existem distribuições que podem induzir que determinados pontos sejam anomalias mesmo que estes na realidade não sejam. Por esse motivo, existem vários

métodos de detecção de anomalias que podem ser utilizados para lidar com as diversas aplicações. Quanto o gráfico à direita os pontos que são classificados como anômalos estão acima de um *threshold* especificado. Eventualmente, há presença de ruídos na série, é o caso do ponto com destaque em verde. Um ruído pode ser definido como um objeto indesejável em um conjunto de dados que não tem relevância, mas que de alguma maneira pode comprometer a análise.

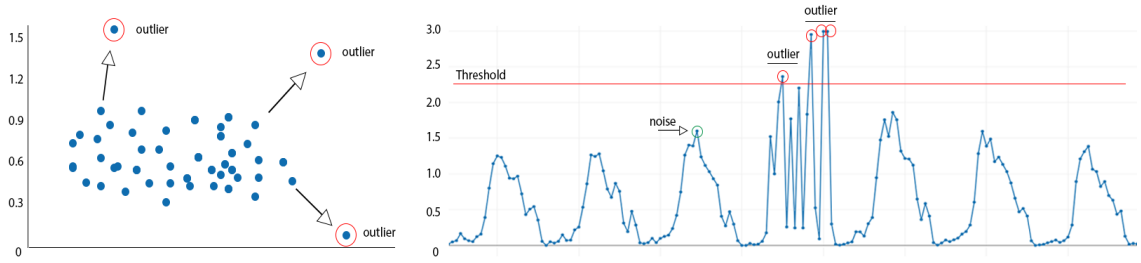

Figura 2 – Representação de dados anômalos em dois formatos de classificação. O formato à esquerda remete a ideia de classificação por grupos. Os pontos mais distantes são classificados como anômalos. No lado direito, temos uma série temporal e os pontos mais distantes do comportamento normal da série são anômalos. O ponto destacado com círculo verde é considerado um ruído.

Em termos práticos, a detecção de anomalias tem se tornado bastante útil pra detecção de fraude, na área de saúde, detecção de intrusos em um sistema, investigação criminal, detecção de erros de medição em dados de sensores e assim por diante. Em cada uma dessas aplicações existe uma maneira para lidar com fronteiras que classificam eventos anômalos ou normais. Isto é, não existe uma abordagem universal para detecção de anomalia, mas sim um conjunto de abordagens adequadas para os mais diversos domínios.

Ademais, existem alguns desafios comuns a muitas abordagens. Por exemplo, definir uma fronteira de decisão entre comportamento anômalo e normal não é trivial e se torna muitas vezes imprecisa. Dessa maneira pontos classificados como anômalos em uma região próxima a fronteira podem na realidade, serem normais e vice-versa. Outro desafio seria definir uma região adaptável que classifique uma região entre normal/anormal mantendo a representatividade no futuro. Ainda outro desafio seria distinguir entre ruído e anomalia uma vez que em alguns dados eles são bem semelhantes (CHANDOLA; BANERJEE; KUMAR, 2009).

1.3 Objetivos

Neste trabalho, nosso objetivo é propor, desenvolver e investigar metodologias que combinem métodos não supervisionados para detecção de anomalia baseado em distância

e modelos supervisionados baseados em aprendizagem de máquina. Avaliamos como a construção dessa metodologia colabora com a construção de um sistema robusto no qual o especialista do domínio terá um entendimento maior em relação ao comportamento das abelhas.

Para que isso seja possível é necessário que algumas etapas sejam realizadas, conforme os itens a seguir:

1. Estudo de métodos não supervisionados de detecção de anomalias.
2. Investigação da série temporal e cadastro de anomalias sugeridas pelo usuário.
3. Investigação de técnicas de janela deslizante para cada um dos algoritmos de detecção de anomalia empregados.
4. Desenvolvimento e avaliação de técnicas de detecção de anomalia não supervisionados com diversos tamanhos de janela.
5. Avaliação final do tamanho de janela por meio da comparação entre os algoritmos de detecção.
6. Desenvolvimento e avaliação de técnicas de aprendizagem de máquina para cada uma das janelas.
7. Comparação e avaliação dos modelos de aprendizagem de máquina empregados.

1.4 Estrutura do Trabalho

Este trabalho está dividido de acordo com os seguintes capítulos:

- Capítulo 2: neste capítulo apresentaremos alguns itens importantes como os tipos de anomalias existentes, alguns métodos e técnicas de detecção de anomalia não supervisionada assim como modelos de aprendizagem supervisionados que serão empregados ao longo deste trabalho tais como *Multilayer Perceptron* (MLP), *Support Vector Machine* (SVM) e *Random Forest* (RF).
- Capítulo 3: neste capítulo apresentaremos alguns trabalhos relacionados a pesquisas recentes em colmeias, como pesquisas que buscam aprimorar conhecimentos sobre fatores que impactam diretamente as abelhas como pesticidas e fatores ambientais. Além disso, são apresentados trabalhos que utilizam dados de sensores para detecção de anomalias em outras aplicações tais como: avaliação de qualidade de dados de redes de sensores *wireless* (WSN), economia de energia em WSN, detecção de eventos anômalos em turbo máquinas dentre outros.

- Capítulo 4: neste capítulo apresentaremos a metodologia empregada neste trabalho como visão geral do trabalho, processo de coleta, pré-processamento, etapa de detecção não supervisionada com o uso dos algoritmos *K Nearest Neighbors* (KNN) e *Local Outlier Factor* (LOF) assim como o emprego dos algoritmos de aprendizagem supervisionada tais como: *Multilayer Perceptron* (MLP), *Support Vector Machine* (SVM) e *Random Forest* (RF).

- Capítulo 5: neste capítulo apresentaremos os resultados alcançados neste trabalho tanto para a etapa de detecção não supervisionada quanto para a aprendizagem supervisionada assim como faremos uma avaliação final dos resultados alcançados nas duas etapas.

- Capítulo 6: neste capítulo finalizaremos com as considerações finais, apresentando uma breve discussão a respeito do trabalho, conclusão, trabalhos futuros e contribuições do autor.

2 Fundamentação Teórica

2.1 Séries temporais

Uma série temporal pode ser definida como uma sequência de observações de uma variável ao longo do tempo (WOOLDRIGE, 2000). Assim, uma série temporal pode ser entendida como uma sequência ordenada de pontos que ocorrem em intervalos de tempo igualmente espaçados. Análise de séries temporais tem sido utilizada em várias aplicações tais como: estudos de utilidade pública, análise de sensos, previsões econômicas, previsões de venda, análise do mercado de ações e assim por diante. Normalmente, uma série temporal é representada por uma sequência de um vetor de observações d-dimensional ordenado: $x(t) = (x_1(t), x_2(t),..., x_d(t))$.

Em geral uma série temporal pode ser decomposta nos seguintes componentes: (1) Componente de tendência: que está relacionado ao comportamento da série em longo prazo. (2) Componente cíclico: que pode ser representado por longas ondas em torno de uma linha de tendência. (3) Componente irregular: que consiste na captura de efeitos que não foram incorporados por outros componentes apresentando flutuações erráticas ou residuais; d) Componente sazonal: flutuações regulares dentro de certo período na série temporal.

2.2 Abordagens para detecção de anomalia

2.2.1 Tipos de anomalias

Anomalias podem ser classificadas nas seguintes categorias:

- Anomalias pontuais: uma única instância de dados é anômala se estiver muito distante do conjunto restante dos dados. É considerado o tipo mais simples de anomalia. Um exemplo de anomalia pontual foi mostrado na figura 2 à esquerda. Pontos que estão distantes de um grupo de pontos são considerados anomalias.
- Anomalias contextuais (ou condicional): a instância de dados é anômala em um determinado contexto específico, mas não necessariamente em outro (SONG et al., 2007). Normalmente é comum em dados de séries temporais, conforme mostra o gráfico na figura 3. As instâncias de dados podem ser definidas a partir de dois conjuntos de atributos:

– Atributos contextuais: são utilizados para determinar o contexto ou vizinhança para cada instância. É o caso de atributos de dados espaciais como latitude e longitude. Em séries temporais o tempo é um atributo contextual que representa a posição de uma instância na sequência de todo conjunto de dados.

– Atributos comportamentais: define características não contextuais de uma instância.

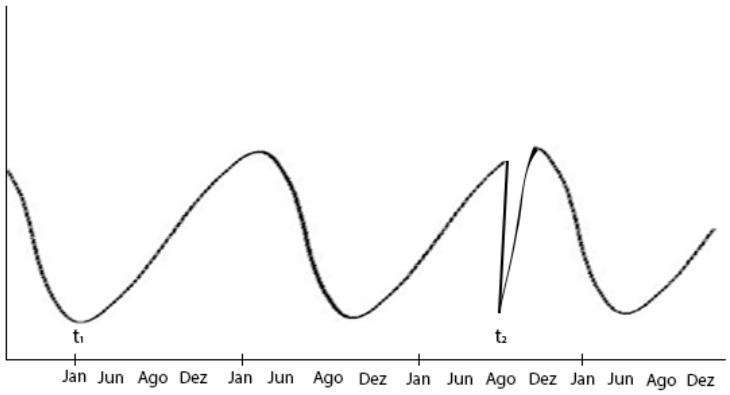

Figura 3 – Exemplo de anomalia contextual, t_1 e t_2 tem o mesmo valor de $_t$ mas ambos ocorrem em diferentes contextos e portanto, não é considerada uma anomalia. **Fonte** (ADAPTADO): (CHANDOLA; BANERJEE; KUMAR, 2009)

- Anomalias coletivas: uma coleção de instâncias de dados relacionadas é anômala em relação a todo o conjunto de dados. Instâncias de dados individuais por si mesmas não podem ser anômalas, mas a ocorrência conjunta como uma coleção é anômala (CHANDOLA; BANERJEE; KUMAR, 2009). Um exemplo de uma série com anomalias coletivas é representado na série da figura 4.

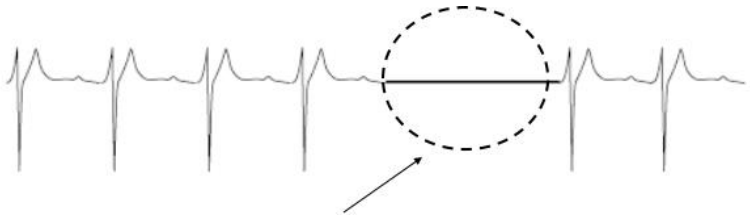

Figura 4 – Exemplo de anomalias coletivas que são representadas por um conjunto de pontos anômalos, em destaque. **Fonte** (ADAPTADO): (SARI, 1986)

A rotulação de dados está relacionada à classificação de cada instância em normal ou anômala. O fato de o conjunto de dados ser rotulado ou não indicará o procedimento de rotulação a ser utilizado. Dessa maneira as técnicas de detecção de anomalia podem trabalhar com as seguintes categorias:

1. Detecção de anomalia não supervisionada: não necessita de dados de treinamento e normalmente são mais aplicáveis.

2. Detecção de anomalia semi-supervisionada: as classes normais já estão ao menos definidas. Não há exigência de anomalias rotuladas.

3. Detecção de anomalia supervisionada: há disponibilidade de conjuntos de dados rotulados para as classes normal e anômala.

Existem dois tipos de saída (*output*) que varia de acordo com a técnica empregada: pontuações (*scores*) e rotulações (*labels*). *Scores* é quando se atribui uma pontuação a cada instância. Essa pontuação representa o grau de anomalia daquele objeto. Por outro lado, uma *label* define uma rotulação categórica para cada instância como, por exemplo: normal ou anomalia.

Existem alguns métodos que podem ser empregados para detectar anomalias em séries temporais e que serão vistos a seguir.

2.2.2 Métodos baseado em estatística

Os métodos baseados em estatística assumem que dados normais ocorrem com elevada probabilidade em regiões de um modelo estocástico, enquanto que anomalias ocorrem em regiões com baixa probabilidade (CHANDOLA; BANERJEE; KUMAR, 2009). É uma abordagem mais simples e baseia-se em rotular instâncias que se desviam estatisticamente de propriedades comuns a distribuição como moda, média, mediana e quartis.

Podem ser divididos em técnicas paramétricas e não paramétricas. A primeira assume uma distribuição prévia estimando parâmetros a partir do conjunto de dados. A segunda não assume nenhuma distribuição previamente.

As abordagens baseadas em estatísticas possuem como vantagem o fato de fornecerem soluções justificáveis para a detecção de anomalia quando as hipóteses assumidas forem verdadeiras. Também, a utilização de um *score* de anomalia fornece uma métrica que vai além de somente rotular dados como normais ou anômalos. O *score* também pode ser usado como intervalo de confiança daquela instância. Além disso, quando se consegue um modelo de estimativa robusto para os dados, as técnicas baseadas em estatísticas podem ser utilizadas também em dados não supervisionados.

Contudo, existem alguns pontos negativos que precisam ser mencionados. Um dos principais é o fato dessa abordagem estar atrelada ao pressuposto de que os dados são gerados a partir de uma determinada distribuição o que normalmente não é verdadeira,

principalmente quando lidamos com conjunto de dados com elevada dimensão. Entretanto, ainda que exista uma suposição razoavelmente justificada com uso de testes de hipóteses a tarefa não é trivial especialmente quando lidamos com distribuições mais complexas. Outro ponto a se destacar é o fato de em que existem situações nas quais dados contenham ruído que possua um comportamento similar ao comportamento anômalo, não existindo uma fronteira de decisão precisa para a separação entre as duas classes, o que acaba sendo uma limitação para esse tipo de abordagem, especialmente para algumas técnicas de média móvel.

A figura 5 mostra um exemplo da técnica de média móvel. Basicamente pressupõe que, se alguma coisa se afasta muito da média então é provável que tenhamos um evento anômalo. Observe que a média em uma série temporal assume valores diferentes e não é uma constante. Isto porque a série temporal é influenciada por diferentes comportamentos do ambiente ao longo do tempo, sofrendo variações por esse motivo a média é móvel (*moving average*).

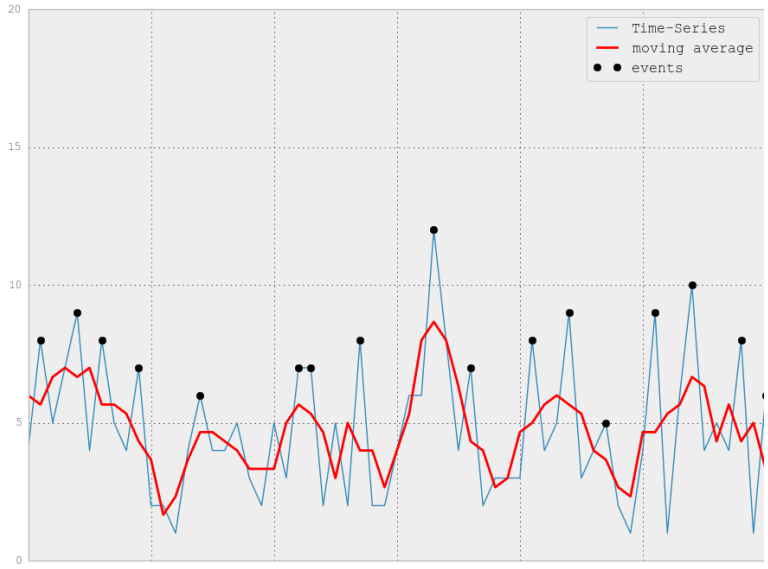

Figura 5 – Exemplo de uma série temporal e a representação da média móvel que assume diferentes valores ao longo do tempo respeitando as mudanças. **Fonte**: (MANSHAEI, 2015)

2.2.3 Métodos de detecção de anomalia não supervisionada

Existem alguns métodos mais populares que são baseados em técnicas de aprendizagem de máquina para detecção de anomalia, conforme veremos nos próximos parágrafos.

2.2.3.1 Detecção de anomalia baseada em densidade

As técnicas baseada em densidade são conhecidas também como baseadas no vizinho mais próximo e parte do pressuposto que dados normais ocorram em uma região mais densa em torno deles, enquanto que anomalias ocorrem em uma região mais esparsa. Existem diversos algoritmos presentes nessa abordagem, mas apresentaremos dois deles: detecção de anomalia global k-NN (*k-Nearest-Neighbor*) e detecção de anomalia local LOF (*Local Outlier Factor*).

2.2.3.1.1 Técnica de detecção global baseada no vizinho mais próximo (k-NN)

O KNN é um algoritmo de detecção de anomalia não supervisionado que se concentra em detectar anomalias globais em um conjunto de dados. É uma maneira simples e direta de classificar dados e utiliza métricas de similaridade, como distância *Euclidiana*, *Manhattan*, *Minkowski* ou *Hamming*.

O algoritmo percorre todas as instâncias e para cada uma destas calcula-se o vizinho mais próximo. O *score* de cada instância é o resultado obtido para cada uma das instâncias utilizando a distância do k $_{\text{th}}$ vizinho mais próximo conforme proposto por (RAMASWAMY; RASTOGI; SHIM, 2000) ou pela média da distância de todos os vizinhos conforme (UPADHYAYA; SINGH, 2012); (ANGIULLI; PIZZUTI, 2002). Normalmente emprega-se um limiar (*threshold*) o *score* de anomalia para delimitar regiões normais de anômalas.

A seleção de um *threshold* apropriado não é trivial, visto que muito depende do conjunto de dados. A escolha do parâmetro **k** é de vital importância. (GOLDSTEIN; UCHIDA, 2016) ilustrou um resultado obtido pelo algoritmo, com k igual 10, conforme mostra a figura 6. Os *scores* podem ser visualizados pelo tamanho da bolha de acordo com a instância correspondente. A cor representa o rótulo onde as anomalias estão marcadas em vermelho. É possível constatar que o k-NN consegue apenas detectar anomalias distantes dos grupos, enquanto que anomalias próximas aos grupos apresentam erroneamente um *score* menor.

2.2.3.1.2 Técnica de detecção local baseada no fator de anomalia local (LOF)

LOF é um dos mais populares algoritmos de detecção de anomalia local e foi proposto por (BREUNIG et al., 2000). A ideia básica por trás deste algoritmo é também atribuir um *score* para uma instância, ao comparar a densidade local de um ponto em relação a densidade de seus vizinhos. A figura 7 mostra que o ponto A é um *outlier* uma vez que a densidade dele é baixa em relação a seus vizinhos.

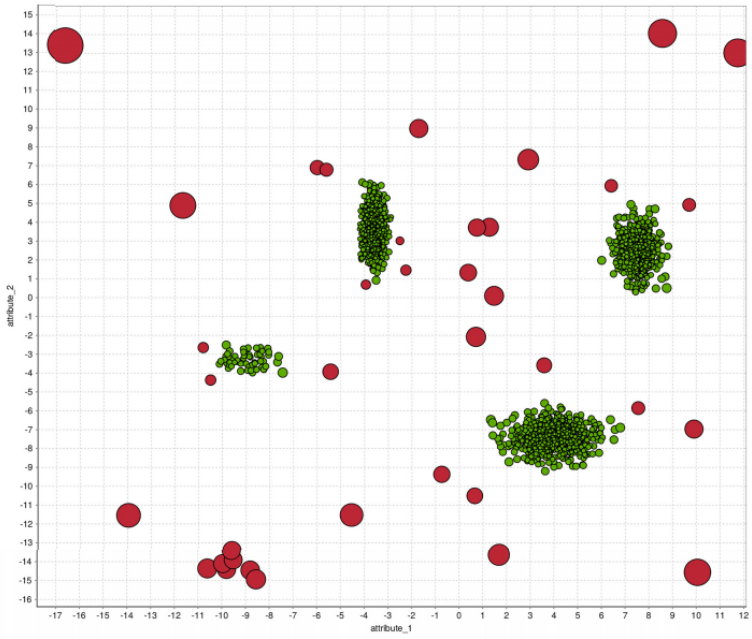

Figura 6 – Visualização dos resultados do algoritmo de detecção de anomalia global. O *score* de anomalia é representado pelo tamanho da bolha enquanto que a cor representa os rótulos. **Fonte**: (GOLDSTEIN; UCHIDA, 2016)

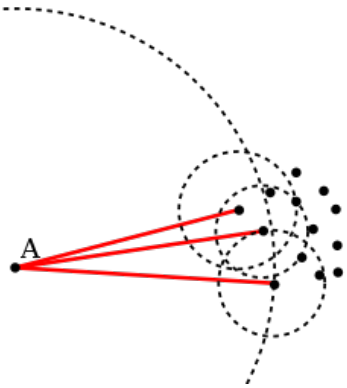

Figura 7 – Exemplo básico do algoritmo. Compara a densidade local do ponto com a densidade de seus vizinhos como pode ser observado pelos círculos em torno dos pontos. **Fonte**: <https://turi.com/learn/userguide/anomaly_detection/local_outlier_factor.html>

1. Para todo registro contido no conjunto de dados encontra-se os k-vizinhos mais próximos.

2. Em seguida, invocamos a máxima distância dos k vizinhos mais próximos da etapa anterior, ou seja, supondo k=4, os 4-vizinhos mais próximos têm distâncias calculadas como 2.3, 9.4, 2.1 e 1.7 logo a distância k selecionada para este ponto é 9.4.

3. Depois, para certo número de pontos (*MinPts*) nós calculamos a distância de acessibilidade (*reach-dist*):

reach-dist$_k$(p, o) = max { k-distance(o), d(p, o) }

4. De posse disso, estima-se a densidade local para **cada ponto** ao calcular a distância de acessibilidade:

$$lrd_{MinPts}(p) = 1/(\frac{\sum_{o \in N_{MinPts}(\mathrm{p})} reach - dist_{MinPts}(p,o)}{N_{MinPts}(p)})$$

5. Por fim, calculamos os *scores* do algoritmo por meio da fórmula:

$$LOF_{\mathrm{MinPts}}(p) = \frac{\sum_{o \in N_{MinPts}(p)} \frac{lrd_{MinPts}(p)}{lrd_{MinPts}(o)}}{N_{MinPts}(p)}$$

O *score* representa a taxa de densidade local dos pontos. Essa propriedade indica que instâncias normais estão em uma região densa (aproximadamente igual a 1) enquanto que anomalias apresentam um valor maior que 1. A configuração do k também é essencial para este algoritmo. O autor (BREUNIG et al., 2000) propõe o uso de uma estratégia de *ensemble* para calcular o LOF. A ideia é computar o *score* para cada k$_{min}$ (ou *MinPts*) até um limite superior (k$_{max}$ ou *MaxPts*) e em seguida selecionar o intervalo k com maior *score*.

2.2.3.2 Detecção de anomalia baseada em *cluster*

No cenário de detecção não supervisionada a *clusterização* aparece como uma das técnicas mais populares (JAIN; DUBES, 1998). Na detecção de anomalia os pressupostos podem ser divididos em três categorias conforme mostra (CHANDOLA; BANERJEE; KUMAR, 2009).

1. Hipótese 1: Instâncias normais pertencem a um *cluster* enquanto que anomalias não estão presentes em nenhum *cluster*. Alguns algoritmos que encaixam nesta hipótese são DBSCAN (ESTER et al., 1996), ROCK (GUHA; RASTOGI; SHIM, 2000) e *WaveCluster* (SHEIKHOLESLAMI et al., 1998). A desvantagem do uso dessas técnicas é o fato de não possuírem um mecanismo para encontrar anomalias, uma vez que o foco de tais técnicas é a formação de *clusters*.

2. Hipótese 2: Instâncias normais estão mais próximos de seus centroides enquanto que anomalias estão distantes.

   Para a construção dessa hipótese duas etapas são necessárias. Primeiramente aplica-se um algoritmo de agrupamento para que os dados sejam agrupados. O segundo passo é percorrer todo o conjunto de dados calculando sua distância para o centroide do *cluster* mais próximo e o *resultado* é o *score* de anomalia. Alguns trabalhos propostos

como o de (SMITH et al., 2002) que estudou técnicas como *Self-Organizing Maps* (SOM), *K-means Clustering* e *Expectation Maximization* (EM) para agrupar dados de treinamento e então utilizar os *clusters* para classificar os dados de teste. Outros trabalhos como o de (BLENDER; FRAEDRICH; LUNKEIT, 1997) foi utilizado para manipulação sequencial de dados. Além de (EMAMIAN; KAVEH; TEWFIK, 2000) que aplicou a mesma hipótese no cenário de detecção de falhas.

A desvantagem dessa hipótese é que se as anomalias formam por si mesmas os *clusters* e as técnicas são capazes de detectar tais anomalias. O surgimento da hipótese 3 foi motivada por essa questão.

3. Hipótese 3: Instâncias normais pertencem a *clusters* grandes e densos enquanto que anomalias pertencem a *clusters* pequenos ou esparsos.

   As técnicas que empregam essa hipótese defendem que instâncias que pertencem a um grupo cujo tamanho e/ou densidade estejam abaixo de um *threshold* são classificadas como anômalas. Alguns trabalhos que se posicionam dentro dessa hipótese pode ser encontrados em (HE; XU; DENG, 2003) que propõe o CBLOF (Fator de anomalia local baseada em *cluster*). O algoritmo obtém o tamanho do cluster em que a instância está presente e calcula a distância da instância para o centro do *cluster*. Outros trabalhos como (CHAUDHARY; SZALAY; MOORE, 2002) e (ESKIN et al., 2002) também segue essa mesma linha.

As técnicas baseadas em *cluster* parecem ser mais robustas, visto que podem ser adaptadas a outros tipos de dados complexos sendo *"plugáveis"* a outros métodos. A eficiência na fase de teste também é rápida, pois computacionalmente o número de *clusters* que serão comparados com cada instância de teste é uma constante. Contudo como aponta (CHANDOLA; BANERJEE; KUMAR, 2009), o desempenho de tais técnicas é altamente dependente do algoritmo de agrupamento utilizado. Como visto, muitas técnicas de detecção de anomalias nessa abordagem não são otimizadas para este fim. Outro fato é que muitos algoritmos de agrupamento forçam para toda instância uma atribuição a um *cluster*, o que pode ser um problema uma vez que anomalias poderiam ser erroneamente atribuídas a grandes *clusters* e como consequência serem consideradas instâncias normais, em casos onde há o emprego de técnicas que pressupõem que anomalias não estão presentes em nenhum *cluster*. Outro fato é o gargalo que pode ocorrer com o uso de alguns algoritmos de agrupamento com complexidade $O(n^2)$.

2.2.3.3 Detecção baseada em Máquina de Vetores de Suporte (SVM)

Uma Máquina de Vetores de Suporte (SVM) introduzido por (VAPNIK, 1995), normalmente utilizada para problemas de classificação, foi adaptada para o problema de

detecção de anomalias. A extensão *One-class* SVM proposta por (SCHöLKOPF et al., 2001) é utilizada para detecção semi-supervisionada e não supervisionada. Isto porque em problemas de classificação semi-supervisionada de classe-única (*one-class*), as técnicas de detecção de anomalia assumem que são fornecidas todas as instâncias de treinamento com apenas uma única classe, conforme pode ser visto na figura 8. Assim as fronteiras discriminativas são aprendidas em torno dessa classe com aplicação de *kernels* robustos em regiões mais complexas.

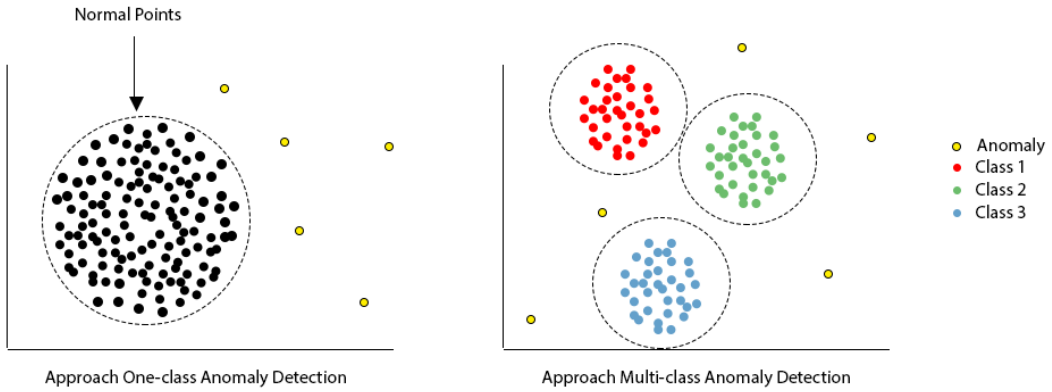

Figura 8 – Idéia básica da diferença entre as duas abordagens.

Assim, o algoritmo é capaz de aprender uma fronteira suave para agrupar instâncias normais usando o conjunto de treinamento e, em seguida, no conjunto de testes as instâncias são identificadas quando estão fora da região aprendida. Dependendo do caso, o *output* pode ser numérico ou textual. No cenário não supervisionado o *One-class* SVM é treinado utilizando um conjunto de dados e, em seguida, atribui-se um *score* para cada instância do conjunto de dados por meio do uso de uma distância normalizada determinada pela fronteira de decisão (AMER; GOLDSTEIN; ABDENNADHER, 2013).

2.3 Aprendizagem supervisionada

Nesta seção veremos algumas abordagens presentes na aprendizagem supervisionada. A aprendizagem supervisionada acontece quando um conjunto de exemplos de entradas e saídas são fornecidos ao classificador. O classificador tem como tarefa aprender a partir daquele conjunto de exemplos.

2.3.1 Perceptron Multi Camadas (MLP)

Perceptron Multi Camadas ou *Multilayer Perceptron* (MLP) consiste numa rede de neurônios simples conhecidos como Perceptron. A ideia central é a retropropagação do

erro entre as camadas intermediárias. Basicamente o Perceptron calcula uma única saída a partir de vários nós de entradas formando uma combinação linear de acordo com seus pesos de entrada e mapeando bem funções não lineares. (ROSENBLATT, 1961)(RUMELHART; HINTON; WILLIAMS, 1986).

A figura 9 apresenta uma visualização do esquema de funcionamento de uma MLP. Primeiramente, o vetor de dados é aplicado na camada entrada e cada neurônio na camada de entrada produz uma saída correspondente. Após isso, os valores são propagados em cada camada até chegar à camada de saída.

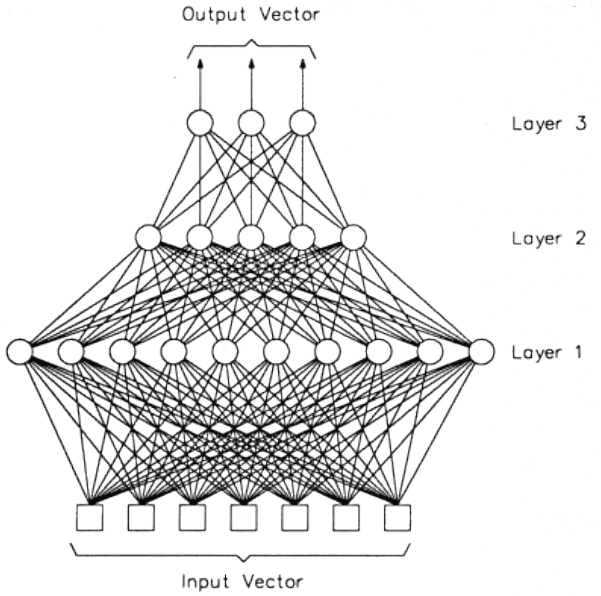

Figura 9 – Representação de uma rede neural multicamadas. Os neurônios são organizados em n camadas e cada ligação entre neurônios tem um peso associado. Os cálculos do vetor de saída atravessam camada por camada. **Fonte**: (LOHNINGER, 2012)

As MLP conseguem manipular tarefas complexas com um grande alcance de aplicações como reconhecimento de voz e imagem. A capacidade de generalização da rede depende do tamanho do conjunto de dados assim como do ajuste adequado dos parâmetros como taxa de aprendizado, *momemtum*, tamanho da rede, conjunto de validação (generalização), etc. Normalmente as MLP convergem com uma boa acurácia. Entretanto, alguns pontos precisam ser levantados como, por exemplo, o modelo "caixa preta"advindo das redes neurais como um todo é um problema para alguns domínios que requerem um entendimento melhor do modelo, além disso, existe também o fato das MLP em algumas situações serem mais lentas em relação a outros algoritmos de aprendizagem. Assim, para algumas aplicações esse ponto pode ser crítico.

2.3.2 Máquina de Vetores de Suporte (SVM)

Máquina de Vetores de Suporte ou *Support Vector Machines* (SVM) representa um classificador binário não probabilístico (embora existam versões com métodos para isso), que visa construir um modelo que mapeia vetores de entrada em um espaço dimensional por meio de um mapeamento não dimensional não definido a priori. Neste espaço uma superfície de decisão linear é construída atuando como um separador de classes com alta capacidade de generalização (CORTES; VAPNIK, 1995). Podem atuar não somente com funções lineares como também não lineares por meio de métodos *kernels* que mapeiam as entradas em um espaço de *features* n dimensional, onde busca-se encontrar um hiperplano ótimo para separar os dados linearmente em duas classes.

As SVMs podem ser divididas em SVMs Lineares e SVMs - Não Lineares. As SVMs Lineares podem ser divididas em SVMs com margens rígidas e SVMs com margens suaves.

As SVMs com margens rígidas têm como característica definir fronteiras lineares a partir de dados linearmente separáveis. A função representa um hiperplano que separa os dados com maior margem, considerando aquele com melhor capacidade de generalização. Assim, há uma definição de uma fronteira linear que divide o conjunto de dados com apenas duas classes +1 e -1. O desafio é encontrar o melhor hiperplano que divida o conjunto de dados, ou seja, que maximize a margem do limiar de decisão.

Por outro lado, as SVMs de margens suaves representam uma extensão das SVMs de margens rígidas, ao lidar com problemas onde dados não são linearmente separáveis, onde é o caso da maioria das situações reais, em que normalmente tem a presença de ruídos e *outliers*. Nesta abordagem não há um hiperplano que divida o conjunto de dados em classes +1 e -1. Ao contrário, a ideia é criar uma flexibilidade em cima das restrições de otimização, utilizando variáveis de relaxamento conhecidas como variáveis de folga. Por sua vez, estas definem o grau de classificação errônea no conjunto de treinamento.

Por fim, as SVMs - Não Lineares possuem como característica mapear o conjunto de treinamento de seu espaço original (não linear), referenciando como entradas, para um novo espaço de maior dimensão conhecido como *feature space*.

A figura 10 mostra a representação de cada uma das abordagens apresentadas. Além disso, o trabalho proposto por (LORENA; CARVALHO, 2007) apresenta mais detalhes teóricos e matemáticos a respeito das abordagens, tratando especialmente de problemas binários.

As SVMs apresentam uma forte base matemática que servem de apoio para a construção das abordagens, isso torna as SVMs capazes de lidar com problemas lineares e não lineares com boa capacidade de generalização. São capazes também de lidar com

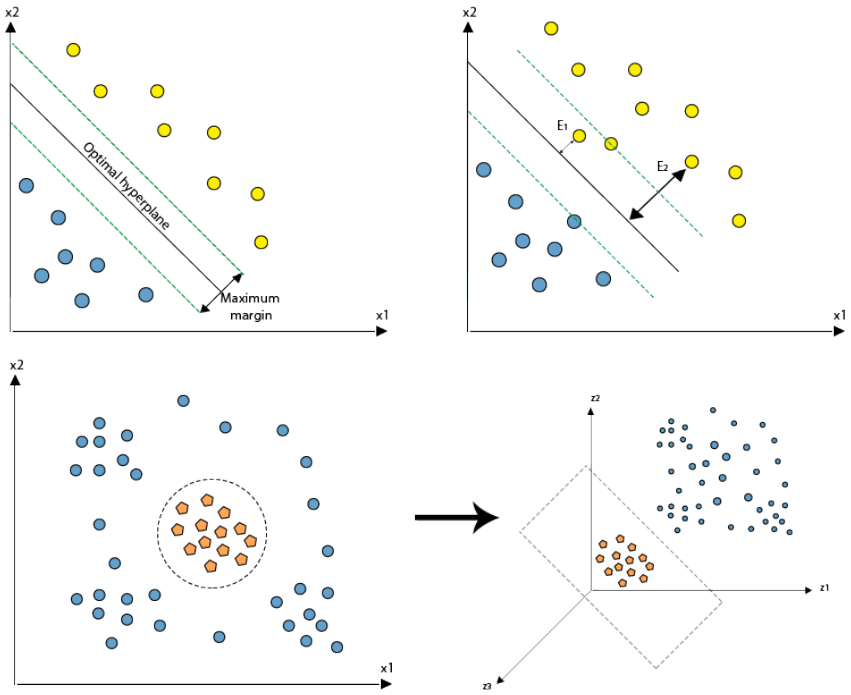

Figura 10 – O gráfico superior à esquerda representa a SVM com margem rígida: a ideia é encontrar um hiperplano que maximize a margem dos dados de treinamento. No lado superior à direita, observamos a abordagem de margem suave, com a presença de E que é a variável de folga, que irá medir onde as amostras se localizam em relação as margens de separação. Os gráficos situados na parte inferior, representam a fronteira não linear no espaço de entradas, (inferior à esquerda), nessa abordagem ocorre a transformação dos dados do R_2 no R_3 (inferior à direita) tornando o problema linearmente separável conforme o gráfico inferior à direita.**Fonte** (ADAPTADO): (LORENA; CARVALHO, 2007)

grande volume dados. O uso de *kernels* em problemas não lineares produz um ganho de eficiência no algoritmo. Contudo, o sucesso das SVM depende muito da escolha dos parâmetros. Além disso, normalmente existe uma dificuldade para interpretação do modelo tornando-o inviável em algumas ocasiões.

2.3.3 Floresta Aleatória (RF)

Floresta Aleatória ou *Random Forest* (RF) foi proposta por (HO, 1995) que considerou que florestas de árvores se separavam em hiperplanos oblíquos, se restringindo apenas a serem sensíveis a um espaço de *features* selecionadas com ganho de precisão sem sofrer *overfitting*. Mais tarde (HO, 1998) ampliou a abordagem ao concluir que outros métodos de divisão, uma vez forçados a serem insensíveis ao espaços de *features* de forma aleatória, tem comportamento semelhante. Devido a alta precisão e redução de *overfiting*

as florestas aleatórias passaram a ser mais difundidas. Basicamente, o RF representa um conjunto de árvores de decisão, que a partir de um vetor de entrada para cada árvore, é possível classificar um objeto e, ao final, realizar uma votação. A classe mais votada é selecionada.

A figura 11 apresenta uma esquema simplificado do funcionamento do RF. São construídas n árvores de decisão com subconjuntos de dados aleatórios. Todos os subconjuntos aleatórios são obtidos a partir do conjunto de dados bruto e também possuem a mesma quantidade de dados. Os dados utilizados nos subconjuntos, após a utilização, são colocados de volta no conjunto de dados total e podem ser selecionadas nas árvores subsequentes. A cada novo subconjunto selecionado o procedimento de *bagging* garante que cada ponto de dados possua a mesma probabilidade de ser selecionado por cada novo subconjunto aleatório. Cerca de 2/3 do conjunto total de dados é inserido em cada subconjunto aleatório e o restante é utilizado exclusivamente para avaliação do modelo *Out-of-bag* (OOB). Após a previsão para cada subconjunto aleatório, é calculada uma previsão final considerando os resultados das previsões individuais. No caso da figura 11, das árvores de 1 a n, a classe "Normal" foi selecionada por maioria de votos.

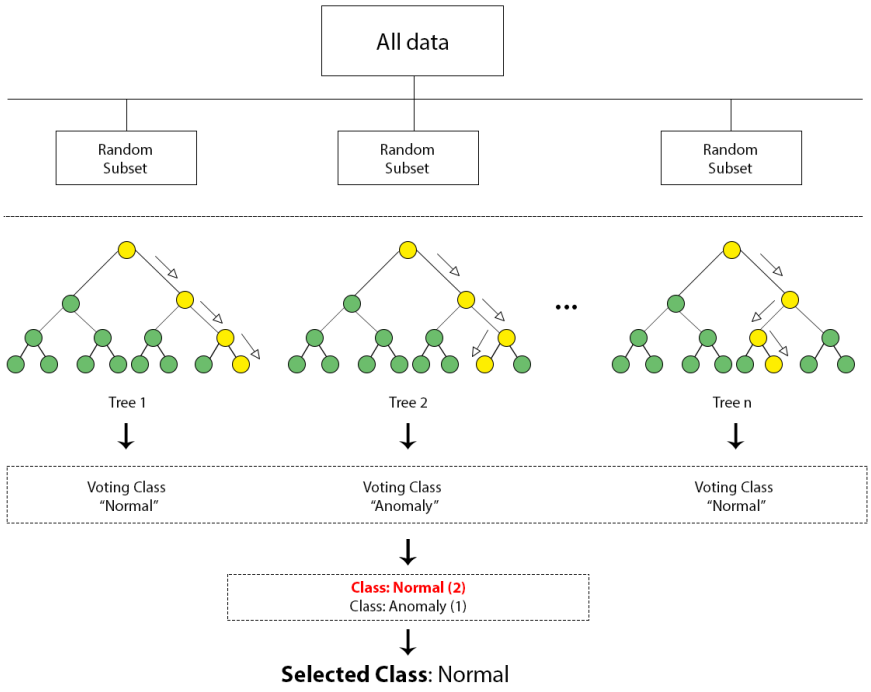

Figura 11 – Esquema simplificado do algoritmo RF. Para simplificar consideramos 3 previsões dos subconjuntos aleatórios. A classe mais votada ("Normal") foi selecionada. **Fonte** (ADAPTADO): (JAGANNATH, 2017)

O mecanismo de *bagging* auxilia na redução do problema de *overfitting* o que é um ponto bastante importante para o RF. Além disso, o RF é eficiente para lidar com grandes conjuntos de dados e com elevado número de *features*. Apresenta elevado poder de

precisão. Também pode ser útil para seleção de *features*, o que torna o RF um dos métodos mais empregados na atualidade. No entanto, a presença de ruídos ainda pode ocasionar *overfitting*. Na etapa de seleção de *features* o algoritmo carece em lidar com variáveis categóricas com diferentes números de níveis, uma vez que se torna mais tendencioso a *features* com maiores níveis, comprometendo a confiabilidade nesta etapa.

Neste capítulo apresentamos os conceitos relacionados à detecção de anomalia não supervisionada e aprendizagem supervisionada fornecendo base teórica para compreensão deste trabalho. No próximo capítulo veremos alguns trabalhos relacionados à detecção de anomalias para análise de dados de abelhas e avaliação de dados em sensores em diferentes aplicações.

3 Trabalhos Relacionados

Um trabalho recente proposto por (COMO et al., 2017) avalia o risco ecológico de produtos de proteção a plantas por intermédio de dados de toxidade aguda de contato. O trabalho apresenta a importância da contribuição ecológica, pois avalia riscos de produtos fito farmacêuticos em abelhas. As abelhas desempenham um importante papel no processo de polinização contribuindo como polinizadores de plantas e culturas selvagens. Para avaliar os riscos dos produtos, os autores aplicaram modelos computacionais para prever o grau de toxidade aguda e crônica, de vários produtos fito farmacêuticos. Para isso, evitando testes em animais, foram utilizados base de dados toxicológicas históricas e modelos quantitativos de relacionamento estrutura-atividade (Q)SAR que foram de grande utilidade para a execução do trabalho. O método computacional aplicado para estimar a toxidade de contato agudo dos agrotóxicos foi o KNN (*K-Nearest Neighbor*). Foram coletados dados de toxidade de contato aguda de diferentes fontes para um total de 256 pesticidas. Dividiu-se em conjuntos de treinamento e teste e os resultados foram satisfatórios. A medida avaliativa do modelo foi à precisão onde os resultados apontaram 70% para todos os compostos e 65% para os compostos altamente tóxicos. Como conclusão, os resultados apontam que o método utilizado foi capaz de estimar, com precisão desejável, a toxidade de pesticidas estruturalmente diferentes e são capazes de rastrear e dar prioridade a novos tipos de pesticidas.

Tendo em mente a importância da abelha como parte vital para a cadeia alimentar como um agente polinizador, (MURPHY et al., 2016) em sua pesquisa utilizou redes de sensores sem fio heterogêneas com o intuito de coletar dados. Foram observados vários parâmetros de uma colmeia para que fosse possível descrever com precisão as atividades e condições internas das colônias. Os parâmetros medidos foram: $C0_2$, 0_2, gases poluentes, temperatura, umidade relativa e aceleração, assim como dados temporais como luz solar, chuva e temperatura. Foi realizada também uma análise biológica para classificar dez importantes estados de colmeia utilizando uma base histórica de implantação em uma colmeia implantada em campo. Essa classificação permitiu a criação de um algoritmo de classificação baseado em árvore de decisão. A análise com os dados de rede de sensores conduziu a uma precisão de 95.38%. Detectou-se também uma correlação entre as variáveis meteorológicas e os dados da colmeia. Essa correlação permitiu a criação de algoritmo para previsão de chuva de curto prazo com base nos parâmetros da colmeia. Dessa maneira, houve contribuição do trabalho para o monitoramento agrícola e ambiental com 95.4% de precisão para previsões locais de curso prazo. Além disso, os resultados também mostraram a eficiência computacional e de energia ao detectar que um nó da rede possa ser utilizado como um dispositivo inteligente autossustentável para colmeias inteligentes.

Ainda no contexto de monitoramento de abelhas (JIANG et al., 2016) propôs a

criação de um sistema de monitoramento automático baseado na tecnologia de redes de sensores sem fio (WSN) para o comportamento das abelhas. O autor considera que a atividade de monitoramento tem que considerar fatores biológicos e físicos dentro e fora da colmeia. O sistema é instalado na entrada da colmeia e pode detectar fatores ambientais externos da colmeia assim como pode fornecer dados de longo prazo das atividades de forrageamento (busca e a exploração de recursos alimentares) de entrada e saída de abelhas com alta resolução temporal. Os resultados mostram quem em média a precisão do comportamento de entrada foi de 84.92% e saída com 85,95%. Os dados de longo prazo (frequências das atividades de chegada e saída somada com os fatores ambientais) foram analisados. Os resultados conduziram a uma conclusão de que as abelhas produtoras de mel tornam-se mais enérgicas quando a média de temperatura torna-se superior a 25º c e a umidade média relativa que ficou entre 60% e 70%.

Em termos de aplicações existem alguns trabalhos focados em tecnologias para tornar as colmeias cada vez mais inteligentes. Por exemplo, o projeto digital de colmeia projetado por (FOTH; BLACKLER; CUNNINGHAM, 2016), permite que apicultores monitorem remotamente a temperatura e umidade nas colmeias assim como permite realizar um controle de peso de abelhas, GPS para localização espacial de colmeias e contagem de abelhas. O esquema também é capaz de interpretar dados de um conjunto de ferramentas que podem avaliar a saúde da colônia.

A detecção de anomalia também foi empregada no trabalho de (ZHANG et al., 2011) que utilizou a detecção de anomalia baseada em estatística para avaliar a qualidade de dados de redes de sensores wireless. A metodologia desenvolvida propôs explorar correlações espaço-temporais em um conjunto de dados existentes em redes de sensores sem fio (WSN) para definir o comportamento normal dos eventos. A metodologia foi testada dentro de um conjunto de dados de WSN disponível gratuitamente em *Grand St. Bernard*, Suíça. Para a análise temporal foram realizados três grandes passos: i) remover a tendência e sazonalidade a fim de alcançar uma série temporal estacionária, ii) ajuste de um modelo de média móvel auto-regressiva (ARMA) para a série temporal estacionária e iii) predição de valores utilizando o modelo ARMA. Para a modelagem de correlação espacial, a análise geoestatística envolveu dois passos principais: i) correlação espacial obtido do cálculo de variograma da amostra e ajuste do modelo e ii) uso do modelo para prever locais não rotulados. As técnicas de detecção de anomalias aplicadas foram: *Temporal outlier detection* (TOD), *Spatial outlier detection* (SOD) e *Spatial-temporal outlier detection* (POD, TSOD e STIOD). Os resultados experimentais concluíram que TOD obteve a menor complexidade de comunicação, porém produziu resultado impreciso. SOD obteve uma elevada acurácia, mas resultou em uma sobrecarga de comunicação. POD e STIOD foram projetados especificamente para reduzir a comunicação, porém eram menos precisos. A técnica preterida foi TSOD que foi capaz de prever dados enquanto

manipula adequadamente anomalias.

Em um mesmo contexto de aplicação (OZDEMIR; UPADHYAYA, 2012) apresentou uma proposta para agregação de dados tolerante a falhas em redes de sensores sem fio de missão crítica mantendo a economia de energia. O ponto chave do trabalho diz respeito ao uso de detecção de anomalia que se baseia na técnica de *hash* sensível à localidade (LSH) que reduz o consumo de energia enquanto elimina dados falsos mantendo a exatidão dos resultados de agregação dos dados com elevada precisão. O trabalho propõe um novo esquema de FTDA (*Fault Tolerant Data Aggregation*) utilizando um mecanismo de detecção de anomalia na rede. O protocolo FTDA consiste em 3 fases: i) coleção de dados e geração de código LSH, ii) detecção de *outlier* e eliminação de dados redundantes e iii) agregação de dados. Os resultados mostraram o sucesso do esquema FTDA utilizando *precision* (onde indica o sucesso do esquema FTDA: verdadeiro positivo), *recall* (taxa de *outliers* reais ou falsos: verdadeiros positivos e falsos positivos) e *F-Measure* (média harmônica entre *precision* e *recall*) como métricas de avaliação. Os resultados comprovaram a capacidade do FTDA de detectar *outliers* em grande parte dos casos e a redução de transmissões de dados falsos aumentando portanto, a precisão de agregação de dados.

Por outro lado, (MARTí et al., 2015) empregou detecção de anomalia no cenário petrolífero especialmente nas máquinas de operação de bombeamento e turbo máquinas a partir da coleta de dados de sensores instalado nas mesmas. Os sensores enviam dados de alta frequência para prevenção de danos e a proposta é fornecer um método robusto para lidar com a escassez de dados rotulados. Para uma detecção eficiente a abordagem realiza uma combinação de um algoritmo de segmentação rápido e de alta qualidade com uma abordagem de *Support Vector Machine* (SVM) de classe única para detecção eficiente de anomalias. O algoritmo de segmentação é o responsável por identificar partes relativamente homogêneas das séries temporais com o objetivo de focalizar a atenção do classificador em partes mais relevantes da série temporal. Assim, partes das séries temporais que permanecem no passado podem ser descartadas com segurança. A partir dos princípios de baixo custo computacional e fácil parametrização foi proposto um novo algoritmo de segmentação chamado que YASA. Os resultados comprovam que o YASA é um algoritmo rápido de segmentação que tem como característica adicional ser parametrizado facilmente e também foi o responsável por identificar seções homogêneas da série temporal dos sensores. As seções são alimentadas para o SVM que cria um modelo de sinais de sensor válidos. Assim, este modelo é usado para detectar situações anômalas nos sensores. Os resultados também comprovaram que a combinação de YASA e SVM superam as outras abordagens.

Ainda no contexto de detecção de anomalias alguns trabalhos foram empregados em diferentes aplicações. O trabalho proposto por (HAQUE; RAHMAN; AZIZ, 2015) propõe uma abordagem que visa distinguir entre condições médicas reais e falsos alarmes,

através da detecção de anomalias de sensores que empregam métodos baseados em predição para comparar e detectar anomalias. Estes métodos de detecção de anomalia estabelecem a correlação espaço-temporal que existe entre os parâmetros fisiológicos. Em um campo mais teórico, (XIE et al., 2011) examina princípios de projeto relacionados às técnicas de detecção de anomalia empregadas nas WSNs (redes de sensores sem fio). São realizadas análises e comparações de abordagens que pertencem a uma categoria técnica similar. Além disso, inclui uma breve discussão sobre áreas de pesquisa que oferecem boas perspectivas no futuro próximo. A detecção de anomalia também é examinada por (GOLDSTEIN; UCHIDA, 2016) que realizou um estudo teórico-prático que incluiu a avaliação de 19 algoritmos não supervisionados para detecção de anomalia, onde revela pontos fortes e fracos das diferentes abordagens. Além disso, há uma investigação de questões como desempenho, esforço computacional, impacto dos parâmetros de configuração, bem como emprego de detecção de anomalia global e local.

3.1 Discussão dos Trabalhos

Neste capítulo apresentamos alguns trabalhos recentes que envolvem pesquisa para o entendimento do comportamento de abelhas e utilização de sensores em diversos contextos. Como visto as pesquisas que estão relacionadas a abelhas, convergiram para o mesmo problema enfrentado: o risco de extinção ou desaparecimento de abelhas, analisando fatores que prejudicam o seu papel como agente polinizador. É o caso do trabalho de (COMO et al., 2017) que avaliou o risco de produtos de proteção a plantas que podem ser altamente tóxicos para as abelhas com a utilização do KNN como método computacional. Adicionalmente, o trabalho proposto por (MURPHY et al., 2016) também revelou preocupação com as abelhas e utilizou árvore de decisão, analisando variáveis climáticas e outros diversos parâmetros meteorológicos para verificar a saúde da colmeia. (JIANG et al., 2016) utilizou redes de sensores sem fio WSN considerando fatores biológicos dentro e fora da colmeia. Outros trabalhos apresentados neste capítulo também aplicam sensores para detecção de anomalias em diferentes aplicações.

Com base nisso, podemos observar que a análise de dados em sensores é algo recorrente. Porém, em se tratando de microssensores implantados em abelhas nosso trabalho se apresenta como uma alternativa de baixo custo e com tecnologia de ponta para monitoramento de insetos. Assim como nos trabalhos citados acima, reconhecemos a importância das variáveis climáticas e consideramos o nível de atividade para analisar o comportamento das abelhas. Além disso, também aplicamos métodos bem difundidos na literatura para detecção de anomalia como KNN e LOF.

4 Metodologia

4.1 Visão geral do trabalho

A figura 12 apresenta um esquema geral deste trabalho, que consiste de duas grandes fases: (1) aprendizagem não supervisionada para detecção de anomalia e avaliação de tamanho de janela e (2) aprendizagem supervisionada para avaliar quais dos modelos utilizados apresentou o melhor resultado.

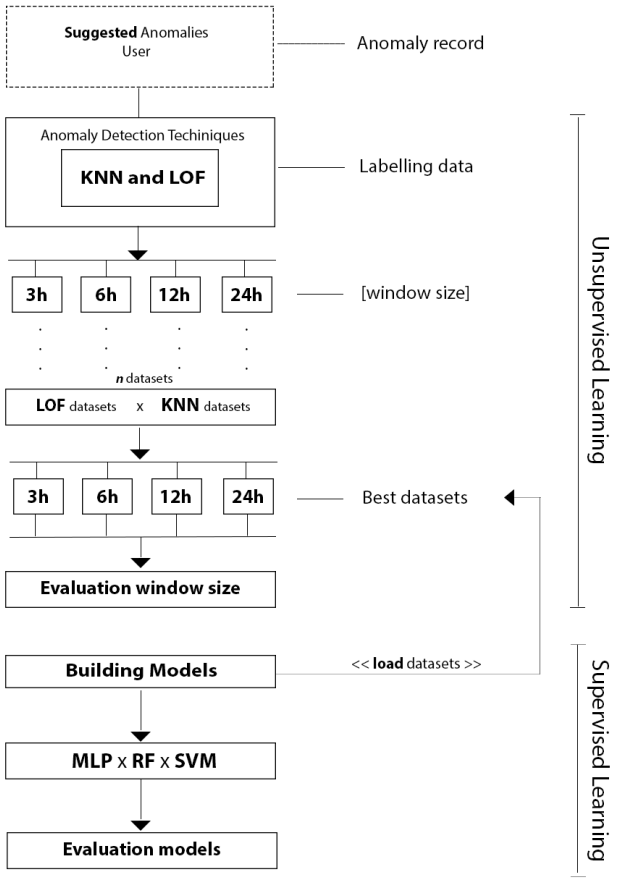

Figura 12 – Esquema geral da metodologia proposta. (1) Cadastro dos pontos anômalos realizados pelo usuário. (2) Aplicação das técnicas de detecção de anomalias. (3) Geração de *datasets* com diferentes valores k para cada janela estudada. (4) Verificação e seleção dos melhores *datasets* por algoritmo. (5) Os melhores *datasets* são selecionados; (6) Avaliação do melhor tamanho de janela na fase não supervisionada; (7) Construção de modelos dos modelos selecionados. (8) Construção dos modelos. (9) Aplicação dos modelos supervisionados (MLP/RF/SVM). (10) Avaliação dos modelos supervisionados.

Na primeira fase ocorre o cadastro dos pontos anômalos sugeridos pelo usuário. Em seguida, iniciando a etapa não supervisionada, aplicamos duas técnicas de detecção de anomalia baseadas em distância: *K-Nearest Neighbor* (KNN) e *Local Outlier Factor* (LOF)

para rotulação dos dados. Assim, para cada técnica empregada, são criados *datasets* com diferentes valores de k em cada janela estudada (3h, 6h, 12h e 24h). Depois comparamos os resultados de ambos utilizando como parâmetro os pontos classificados pelo usuário e assim extraímos as métricas. Os melhores *datasets* são selecionados e avaliados. As métricas extraídas permitiram também selecionar o melhor tamanho de janela.

Na etapa supervisionada empregamos métodos de aprendizagem de máquina utilizando como entrada os melhores *datasets* sugeridos na fase anterior. Empregamos e avaliamos três modelos para nosso estudo: *Multilayer Perceptron* (MLP), *Random Forest* (RF) e *Support Vector Machine* (SVM). Assim, encerramos com a última etapa da nossa metodologia ao realizar a avaliação dos modelos empregados.

Neste capítulo abordaremos aspectos práticos relacionados à pesquisa realizada para o desenvolvimento da metodologia para detecção de anomalias em dados de abelhas.

4.2 Coleta de dados

O processo de coleta foi realizado através de um sistema instrumentado conforme mostra a figura 13. A espécie de abelha *Melipona fasciculata* foi analisada, junto a Embrapa – Amazônia Oriental (Belém-PA-BR). No canto inferior a presença da abelha (*M. fasciculata* nativa da região amazônica) com etiqueta de RFID presa junto ao tórax. O período de coleta foi entre 1 a 31 de Agosto de 2015 onde afixamos etiquetas eletrônicas RFID (*Radio-Frequency Identification*) em um total de 1280 abelhas sendo 40 abelhas por colmeia a cada semana. O sistema instrumentado detecta o momento que uma abelha passa pelo leitor RFID e, neste momento, registra-se um movimento. Neste período registrou-se um total de 127.758 atividades o que significa aproximadamente 100 registros de atividade por abelha foram detectados durante todo o experimento.

Como vimos, para calcular o nível de atividade das abelhas no sistema basta dividirmos o número total de movimentos em um dado período, pelo número de abelhas vivas naquele instante. Para ilustrar, considere que quando o valor do nível de atividade for 0.0 significa que nenhuma abelha está realizando atividade naquele momento se, por exemplo, for 1.0 significa dizer que cada abelha está realizando em média uma única atividade naquele momento. Durante o experimento havia em média, entre 240 e 320 abelhas vivas por dia.

Figura 13 – Esquerda: (1) Colmeia da abelha *Melipona fasciculata*, (2) Intel Edison para controle do RFID e armazenamento de dados, (3) caixa de PVC para armazenamento dos itens eletrônicos, (4) leitor de antena RFID, (5) tudo plástico para a passagem das abelhas. Topo à direita: Visão geral das 8 colmeias. Inferior à direita: Abelha com etiqueta RFID anexada ao tórax.

4.3 Pré-processamento

Dentre as variáveis coletadas, selecionamos a *feature* de nosso interesse: nível de atividade. No conjunto de dados bruto, percebemos a presença de muitos ruídos e poucas anomalias. Assim decidimos gerar sob a base de dados original uma perturbação no conjunto de dados a fim de forçar o surgimento de mais anomalias (peça principal de investigação). Assim este novo conjunto de dados foi utilizado para as próximas etapas.

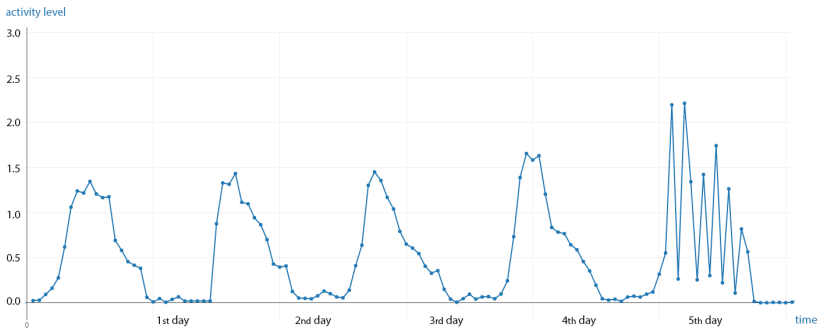

Figura 14 – O gráfico ilustra os 5 primeiros dias do comportamento acumulado das colmeias.

Os aparentes ruídos presentes na série foram mantidos para que pudéssemos verificar a efetividade dos métodos empregados para detecção de anomalia. A figura 14 ilustra a série temporal do nível de atividade de abelhas dos 5 primeiras dias de análise.

Podemos notar a presença de alguns pequenos picos que sugerem a presença de ruídos. A escolha de um método que possa distinguir entre pontos ruidosos e anômalos não é uma tarefa trivial. Além disso, notamos a presença de eventos anômalos na colmeia por volta do 5º dia, provocados por algum fator. Nos dias anteriores, podemos identificar padrões comportamentais dentro do nível de normalidade nas abelhas. Costumeiramente as abelhas apresentam atividade mais elevada durante o período de presença solar.

4.4 Etapa de detecção não supervisionada

Nesta etapa aplicamos os métodos de detecção de anomalia baseados em distância. Para a execução desta etapa utilizamos o R 3.4.1 *version* para a criação/execução dos scripts de LOF e KNN [1] para a etapa não supervisionada. Antes do início desta etapa, um usuário registra os pontos anômalos da série temporal na base de dados resultando em um total de 75 pontos anômalos sugeridos. A ideia é confrontar as anomalias sugeridas pelo usuário com as anomalias sugeridas pelos algoritmos de detecção e obter uma métrica para avaliação dos algoritmos.

O inicio desta etapa ocorre quando empregamos dois algoritmos não supervisionados para detecção de anomalia o *K Nearest Neighbors* (KNN) e Local Outlier Factor (LOF). A figura 15 retrata o funcionamento do conceito de janela deslizante na série temporal para ambos algoritmos. Para classificar um ponto como anômalo é necessário considerar seus k vizinhos. Por exemplo, para verificar se o ponto vermelho é uma anomalia em uma janela de tamanho 6 com k igual a 5 é necessário considerar os 5 vizinhos da janela (pontos amarelos). Este processo é realizado ao longo de toda série com deslocamento da janela ponto a ponto.

O KNN é um algoritmo que tem como capacidade detectar anomalias globais em um conjunto de dados. Assim, para todo ponto contido na base, os k vizinhos mais próximos devem ser encontrados. A partir do cálculo de distância dos vizinhos mais próximos, é obtido um *score* de anomalia que corresponde a média da distância dos vizinhos em torno daquele ponto como estimativa para densidade(ANGIULLI; PIZZUTI, 2002)(UPADHYAYA; SINGH, 2012). Utilizamos a distância euclidiana para cálculo da distância entre os pontos. Além disso, pontos calculados com *score* superior a um *threshold* previamente definido, são classificados como *outlier*. A seleção de um *threshold* adequado depende da distribuição da densidade dos pontos, além da verificação e ajuste do número de anomalias detectáveis pelo algoritmo. Pela figura 16 notamos que pontos alocados em uma região mais esparsa são classificados como *outlier*. Utilizamos um *threshold* para delimitar regiões mais densas de regiões esparsas. O grau de *outlierness* que mensura

[1] Os *scripts* desenvolvidos podem ser acessados em: <https://github.com/ffgama/swarmsensing/> ou podem ser vistos no Apêndice A

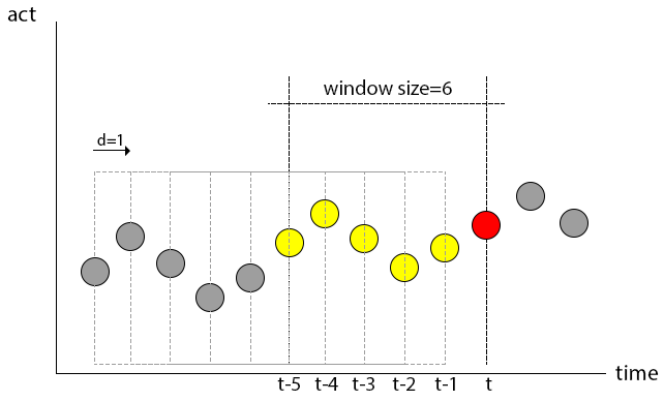

Figura 15 – A figura ilustra o funcionamento o funcionamento dos algoritmos em uma janela deslizante com deslocamento d igual a 1. O ponto vermelho indica o ponto que será predito que toma como base os pontos amarelos.

o nível de anomalia do ponto, é representado pela variação das cores que, quanto mais escuras, maior o *outlierness*. Este procedimento é adotado não somente no KNN como também no LOF. Podemos perceber que os pontos que estão acima do *threshold* definido como 1.8 aproximadamente, podem ser como considerados anomalias globais. Para o KNN, em cada janela realizamos testes para seleção do melhor k como mostra a tabela 1.

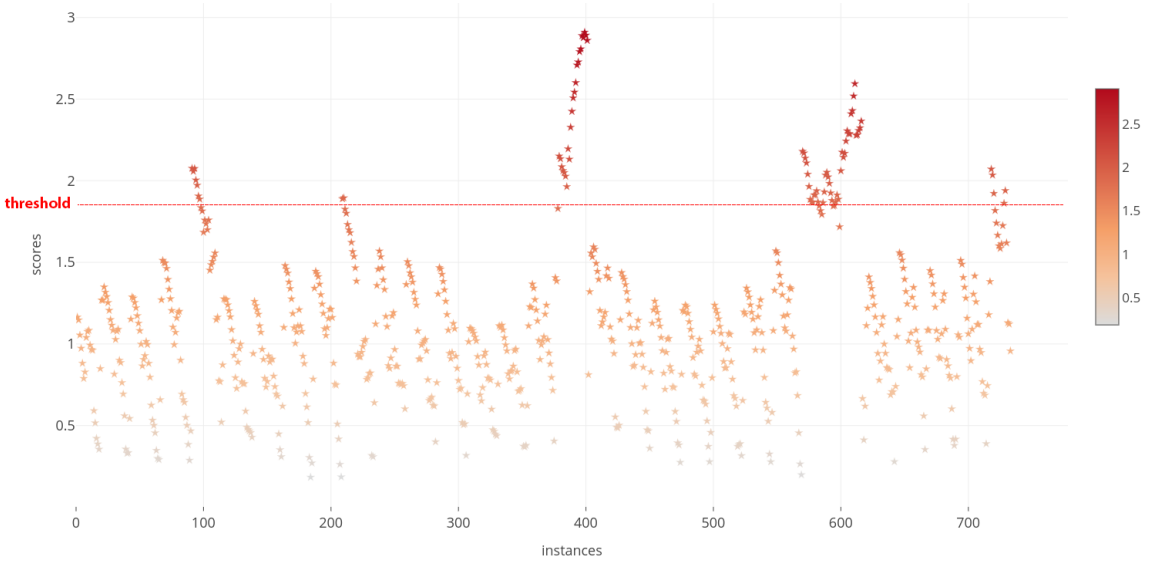

Figura 16 – Distribuição das densidades dos pontos do KNN para a janela de tamanho 12 com valor de k=11 e threshold 1.84. Foram detectados 81 pontos anômalos. Pontos mais escuros representam o grau de *outlierness.*

Além do KNN aplicamos outro algoritmo de detecção com uma perspectiva voltada para detecção local: o LOF (BREUNIG et al., 2000). Basicamente, o algoritmo mostra

como resultado um *score* que indica o valor *outlier*. *Scores* com valores próximos de 1 indicam que o ponto pertence ao *cluster* e está dentro de uma região de densidade homogênea. Em contraste, valores de densidade distantes de 1, sugerem que o ponto está em uma região esparsa em comparação com seus vizinhos. Nesse último caso o ponto pode ser considerado um *outlier*.

Tabela 1 – Um resumo dos casos de teste para determinar o valor de k. Os intervalos para configurar os valores de k são determinados para diferentes tamanhos de janela. As células em destaque representam os intervalos selecionados para cada janela.

Casos de teste para o Algoritmo KNN			
3h	6h	12h	24h
k1	k1	k1	...
k2	k2	k2	k10
	...	...	...
	k5	k7	k15
		...	...
		k11	k20
			...
			k23

Ao realizar os cálculos de densidade do algoritmo, é razoável escolher um bom intervalo para k, que varia de um valor mínimo a um máximo que é definido pelo usuário. De acordo com (BREUNIG et al., 2000), não é recomendável usar um valor muito pequeno para k tendo em vista o surgimento de flutuações estatísticas indesejáveis. A tabela 2 mostra os casos de teste resumidos para cada janela. Os intervalos foram testados em relação aos limites do tamanho máximo dos vizinhos em relação à janela de tempo (tamanho da janela menos 1). Por exemplo, a janela 12h exibe o primeiro conjunto de teste com um valor mínimo de k igual a 1 e máximo de k igual a 11.

Tabela 2 – Um resumo dos casos de teste para determinar o valor de k. (:) representa sequência de k, por exemplo, k1:k5 é equivalente a k1 até k5. Os testes são realizados para cada janela. As células em destaque representam os intervalos selecionados para cada janela.

Casos de teste para o Algoritmo LOF			
3h	6h	12h	24h
k1:k2	k1:k5	k1:k11	k1:k23
	k2:k5	k2:k10	k10:k20
	k3:k5	k3:k9	k11:k20
	k4:k5	k5:10	k12:k20
		k6:k10	...
			k16:23

Aplicamos o algoritmo LOF que gerou um *score* para cada uma das 744 instâncias do conjunto de dados em cada caso de testes exibidos na tabela. Para os intervalos de k, obtivemos o valor de densidade para cada instância para cada k. Adotamos uma heurística que foi selecionar o valor máximo do intervalo. Por exemplo, no caso da janela k1: k5, obtivemos 5 valores de densidade (*score*) para essa instância e o valor mais alto entre estes 5 é selecionado. Depois de rotular os dados para todas as instâncias, os pontos classificados pelo algoritmo foram comparados com os pontos classificados pelo usuário, seguindo o mesmo procedimento para todos os casos de teste restantes. Finalmente, o melhor intervalo k foi escolhido para cada janela temporal de acordo com a maior taxa de acertos do número total de pontos sugeridos pelo algoritmo como anomalias, em relação aos pontos sugeridos como anômalos pelo usuário.

De posse dos 8 *datasets* gerados da etapa anterior, começamos a próxima fase da etapa não supervisionada. Dessa maneira, realizamos a comparação entre os algoritmos de detecção (KNN x LOF) para cada janela temporal a fim de selecionar os melhores *datasets*.

Os resultados dessa investigação entre KNN e LOF e da investigação entre as janelas serão apresentados na seção 5.1.

4.5 Etapa de aprendizagem supervisionada

Na etapa supervisionada construímos modelos utilizando os melhores *datasets* selecionados na etapa anterior. Investigamos a capacidade de aprendizado de 3 modelos supervisionados baseados em aprendizado de máquina: *Multilayer Perceptron* (MLP), *Support Vector Machine* (SVM) e *Random Forest* (RF).

Para a construção dos modelos utilizamos o *Weka Experimenter*[2] para configurar os experimentos, dando autonomia para execução de todos os modelos e ao final obter um valor médio, a partir de um número de execuções previamente definido, no nosso caso, 30 execuções. Ao final das execuções obtêm-se várias métricas avaliativas, dentre elas acurácia, precisão e roc.

Em nosso experimento utilizamos a média de cada métrica por meio da classe (*AveragingResultProducer*) presente no Weka, para comparar os modelos. Empregamos validação cruzada (*CrossValidationResultProducer*) com k igual a 10-*folds*. Onde em cada execução o conjunto de dados é dividido em subconjuntos mutuamente exclusivos separando 90% para o conjunto de treinamento e 10% para o conjunto de teste ao longo

[2] Os resultados de cada modelo podem ser encontrados no repositório: <https://github.com/ffgama/swarmsensing/tree/master/scripts/extra/supervised_results>

de 10 iterações. Ao final deste processo obtêm-se métricas para aquela execução. Os resultados finais são salvos em *csv* e exibidos em um gráfico.

Para a seleção dos parâmetros realizamos alguns testes manualmente. Repetidamente, fizemos ajustes e avaliamos a acurácia do modelo. Os parâmetros dos modelos são apresentados na tabela 3.

Tabela 3 – Configuração dos parâmetros dos modelos. Esses parâmetros foram aplicados para todas as janelas temporais estudadas.

MultilayerPerceptron	Value
HiddenLayers	1,2
LearningRate	0.3
Momentum	0.2
Epochs	500

SupportVectorMachine	Value
C	1
Degree	3
Gamma	0

RandomForest	Value
NumberIterations	400
NumberAttributes	0
TrainingSet	100

Para o MLP selecionamos (1,2) camadas ocultas com taxa de aprendizagem de 0.3, *momemtum* 0.2 com um total de 500 iterações. Em relação ao SVM utilizamos a biblioteca SVMLib[3](CHANG; LIN, 2011) e configuramos para o parâmetro C o valor 1 o que representa o tipo de SVM empregado: *nu*-SVC (*nu* - Support Vector Classification) e onde *nu* é um parâmetro adicional para controle do número de vetores de suporte. *Degree* com valor 3 representa a função kernel *sigmoid* e *Gamma* com valor 0. Por fim no RF para o número de iterações foi de 400, número de atributos aleatórios que serão investigados 0 e porcentagem do tamanho do conjuto de treinamento para cada tamanho de *bag* igual a 100.

Os resultados dessa investigação entre os modelos supervisionados: MLP, SVM e RF, serão apresentados na seção 5.2.

Neste capítulo apresentamos a metodologia proposta deste trabalho bem como as ferramentas utilizadas. No capítulo seguinte serão apresentados os resultados alcançados.

[3] Mais detalhes também podem ser vistos no endereço: https://www.csie.ntu.edu.tw/ cjlin/libsvm/

5 Resultados

5.1 Etapa de detecção de anomalia não supervisionada

Nesta etapa avaliamos os algoritmos empregados: KNN e LOF em relação ao tamanho das janelas. Os resultados dos algoritmos são confrontados com as anomalias sugeridas pelo usuário. O objetivo é encontrar qual algoritmo que apresenta os melhores resultados em termos de acurácia.

A partir do gráfico 17 nota-se que os 8 melhores *datasets* de cada algoritmo são comparados janela a janela. Pelo gráfico, percebemos que em todas as janelas o KNN foi superior, com destaque para a janela de tamanho 3 que alcançou aproximadamente 95% de acurácia. Quanto ao LOF o melhor resultado foi observado na janela tamanho 24 com aproximadamente 83%. Assim, todos os *datasets* selecionados para as próximas etapas foram: KNN-3, KNN-6, KNN-12, KNN-24.

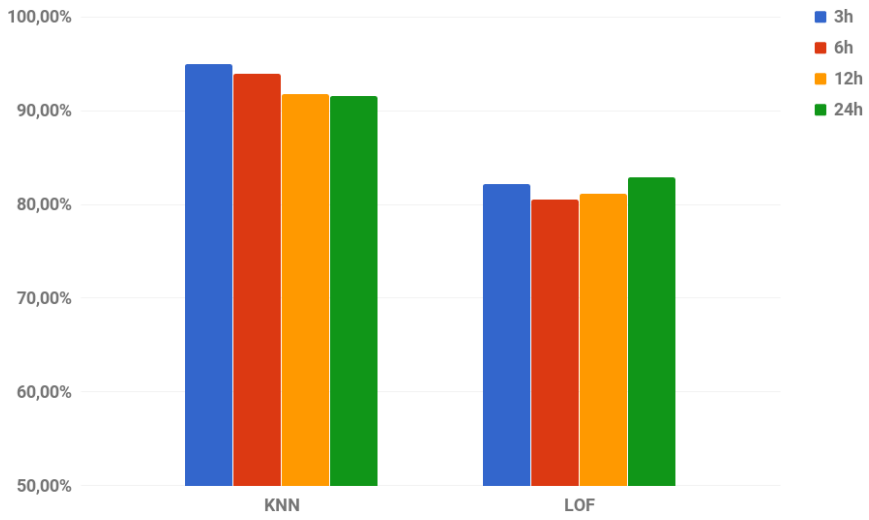

Figura 17 – Gráfico comparativo entre KNN e LOF. No KNN, observamos que a acurácia decresce à medida que aumentamos a janela. Quanto ao LOF não existe essa relação inversa, o melhor resultado encontrado foi na janela de tamanho 24.

Finalmente na última fase da etapa não supervisionada, avaliamos os 4 melhores *datasets* resultantes por intermédio de uma de matriz de confusão de acordo com a figura 18. Assim podemos avaliar os melhores *datasets* em um maior nível de granularidade. Assumindo que a classe positiva seja "normal" percebemos que TN decresce à medida que o tamanho de janela aumenta. Em contrapartida, notamos uma relação direta entre o número de erros (FN e FP) e o tamanho da janela. Portanto, para a etapa não supervisionada a

janela de tamanho 3 foi a que obteve maior taxa de acerto e portanto foi avaliada como a mais adequada para lidar com nosso problema.

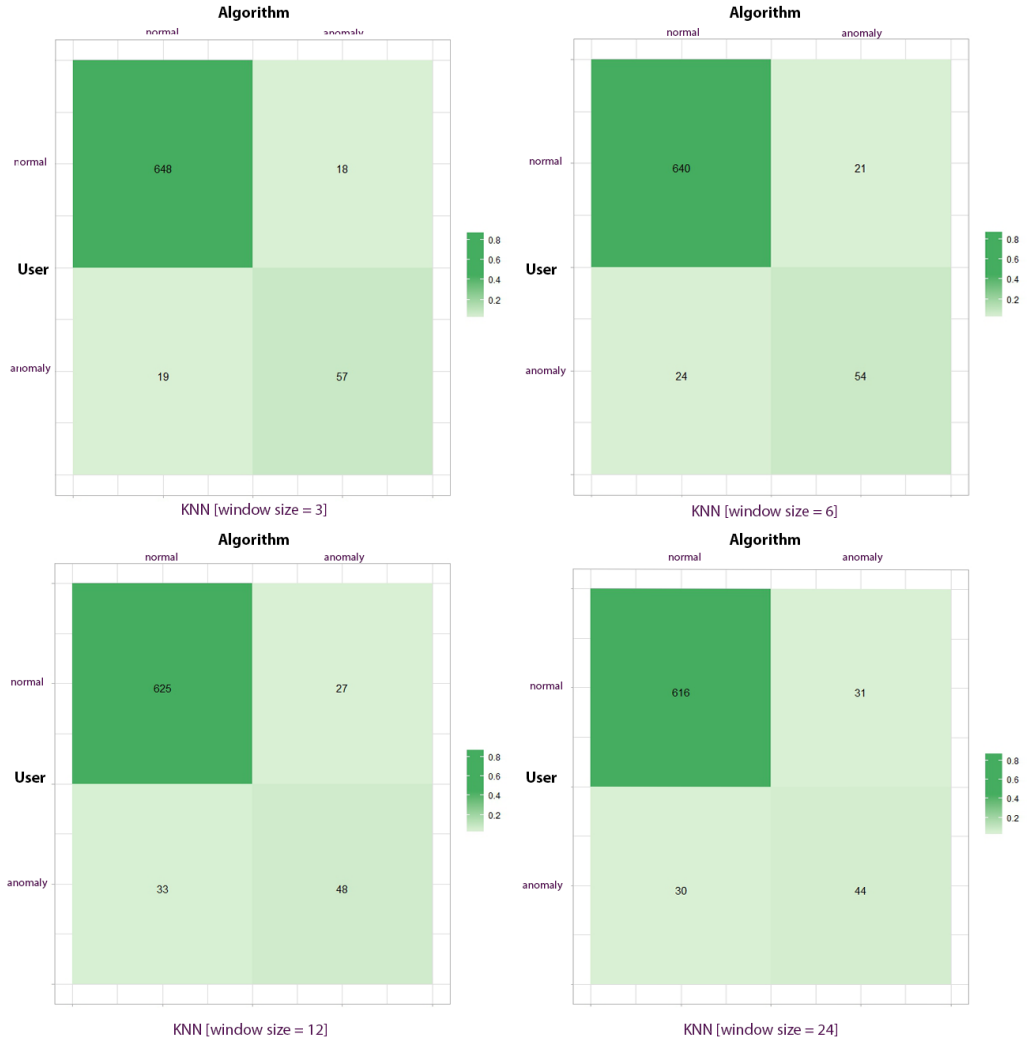

Figura 18 – A matriz de confusão exibe os melhores *datasets* após a comparação KNN x LOF. Observa-se que a região TN (quando ambos, usuário x algoritmo concordam que há anomalia) vai ficando mais clara à medida que o tamanho da janela aumenta os erros também vão aumentando na mesma proporção.

5.2 Etapa de aprendizagem supervisionada

Nesta etapa avaliamos os modelos empregados: MLP, SVM e RF utilizando os melhores *datasets* da etapa anterior. O objetivo é identificar qual o modelo que melhor se ajusta ao conjunto de dados.

Os resultados alcançados na etapa supervisionada são apresentados no gráfico na figura 19. A partir do gráfico notamos que em geral, a janela tamanho 3h obteve melhores resultados com 92.26%, 89.73% e 92.67% para cada modelo, respectivamente. Além

disso, notamos que para *Support Vector Machine* as janelas 3h e 24h obtiveram melhores resultados e praticamente a mesma acurácia 89.73% e 89.74%, respectivamente. Quanto ao *Multilayer Perceptron*, as janelas 6h ,12h, 24h obtiveram resultados próximos: 90.99%, 90.12%, 90.65%, respectivamente. No Random Forest, as janelas 6h e 12h apresentaram taxas aproximadas, respectivamente: 90.60% e 90.73%.

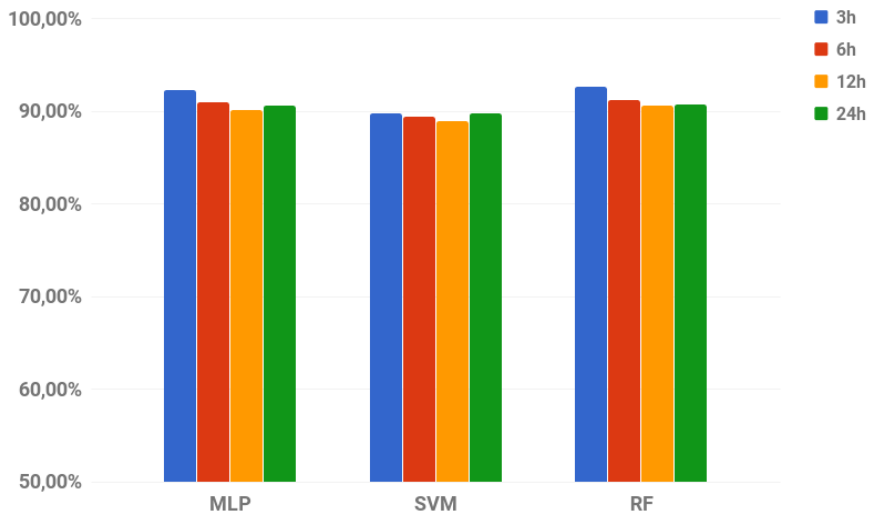

Figura 19 – Comparação entre os 3 modelos supervisionados, onde o RF apresentou os melhores resultados em relação aos outros modelos.

Tendo em vista a próxima etapa realizamos a avaliação dos modelos para verificar quais deste obteve os melhores resultados. Assim, observamos que o *Random Forest* teve taxas superiores em todas as janelas, na ordem de: 92.67%, 91.20%, 90.60%, 90.73% de acurácia.

O gráfico na figura 20 apresenta os resultados obtidos pelo RF e KNN para cada janela considerando: acurácia, precisão e ROC.

Na seção superior, notamos que no RF o tamanho da janela 3h foi superior em todas as métricas: precisão de 93.60%, acurácia de 92.67% e ROC de 87.10%; seguido pela janela de tamanho 6h que apresentou resultados de precisão 91.60%; acurácia de 91.20% e; ROC de 79.20%. As janelas 12h e 24h obtiveram resultados bem próximos para precisão e acurácia, porém, a janela 24h obteve uma ligeira vantagem na medida ROC 77.40% contra 73.80%.

Para o KNN, na seção inferior do gráfico, o tamanho da janela 3h obteve os melhores resultados em todas as métricas de avaliação com resultados de: 95.01% para acurácia, 86.58% de ROC e 76% de precisão; seguido pela janela de 6h com 93.91% de acurácia, 84.19% de ROC e 72% de precisão; seguido pela janela de 12h com 91.81% de acurácia, 79.49% de ROC e 64% de precisão e; a janela 24h que obteve os piores resultados

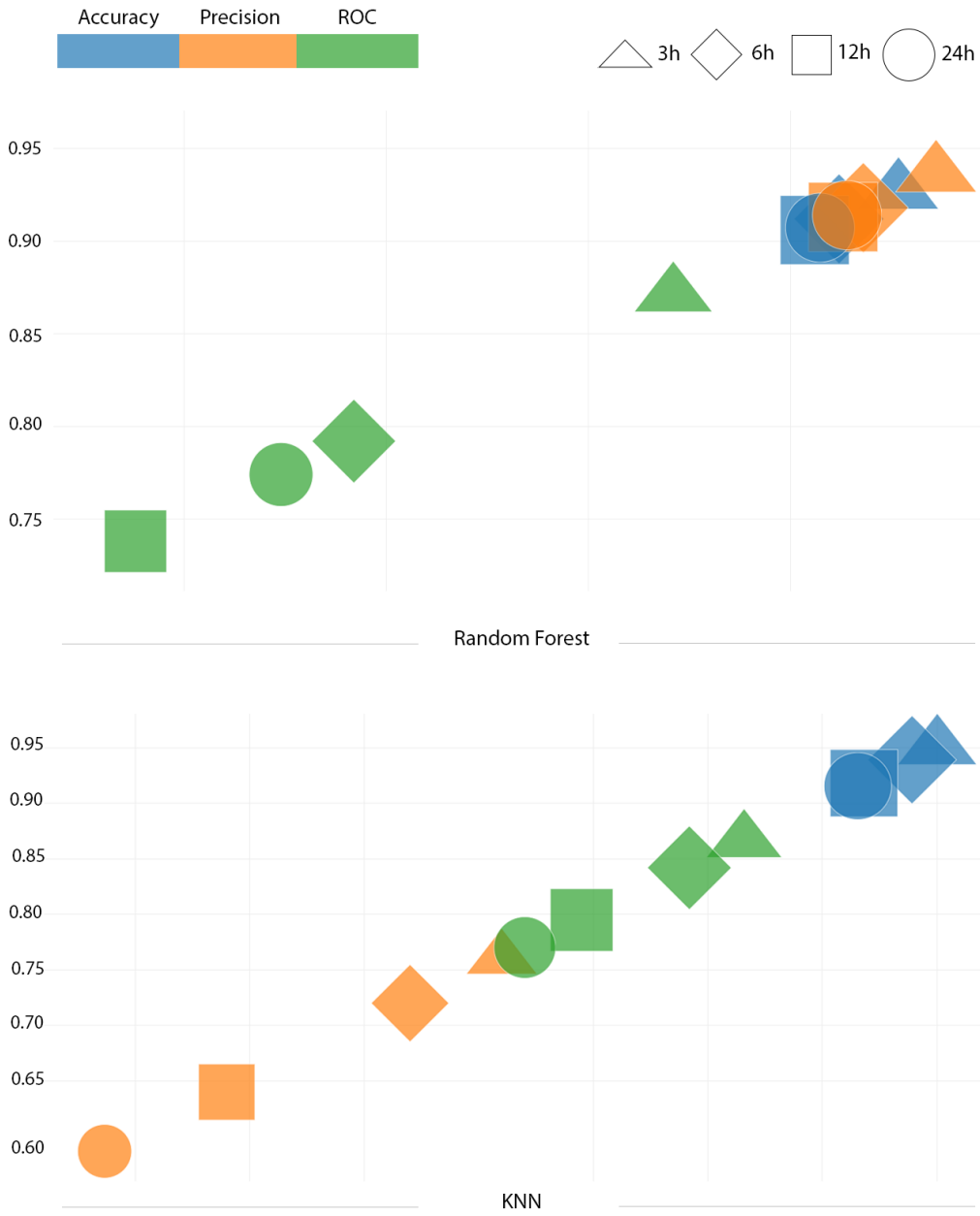

Figura 20 – Comparação entre as medidas de avaliação (acurácia, precisão e roc) dos modelos selecionados. No RF, a acurácia e a precisão obtiveram resultados bem semelhantes especialmente nas janelas 12h e 24h, enquanto que no KNN os resultados obtidos para cada uma das janelas estão mais agrupados.

com 91.54% de acurácia, 77,01% de ROC e 58,67% precisão.

Neste capítulo apresentamos os resultados obtidos após o emprego de nossa metodologia. No próximo capítulo encerraremos nosso trabalho com as considerações finais.

6 Considerações Finais

6.1 Discussão

De modo geral, os resultados alcançados neste trabalho foram satisfatórios. A comparação do KNN e LOF mostrou que um método de detecção global acoplado com o mecanismo de janela deslizante se revelou mais adequado para o nosso problema. Ademais, a utilização de métodos supervisionados bem difundidos nos trabalhos científicos como *Multilayer Perceptron*, *Support Vector Machine* e ROC conduziram a uma análise mais profunda da nossa metodologia.

Na etapa não supervisionada, notamos que o KNN apontou que à medida que aumentamos o tamanho da janela, menor se torna a capacidade de detecção do algoritmo. Por outro lado, essa relação inversa não foi detectada no LOF. Acreditamos que, os resultados do LOF tenham sido inferiores devido sua alta sensibilidade em detectar anomalias em pontos ruidosos.

Adicionalmente, dentre os três modelos empregados: MLP, SVM e RF, este último, obteve taxas superiores em todas as janelas com acurácia de 92.67%, 91.20%, 90.60%, 90.73%, respectivamente. Também fizemos o uso de métodos de reamostragem como o *cross-validation* tendo em vista a redução de *overfitting*. Percebemos também um ganho significativo de precisão em todas as janelas nesta etapa.

6.2 Conclusão

Neste estudo, foi proposto, desenvolvido e investigado metodologias que combinaram métodos de detecção não supervisionados baseados em distância como *K Nearest Neighbor* (KNN) e *Local Outlier Factor* e modelos de aprendizagem supervisionada como *Multilayer Perceptron* (MLP), *Support Vector Machine* (SVM) e *Random Forest* (RF).

Na etapa de detecção não supervisionada a escolha do k percebemos que este foi determinante para selecionar os melhores *datasets*. Para o KNN, concluímos que, em geral, para cada tamanho de janela a proporção de acertos foi maior quando utilizamos um número máximo de vizinhos suportados naquela janela. Para o LOF, percebemos que a utilização de intervalos máximos para as janelas 3h, 6h, 12h se mostrou mais adequada enquanto que para a janela 24h a seleção do melhor intervalo esteve diretamente relacionada à escolha de um intervalo razoável no número de vizinhos.

Quando comparados, os resultados mostraram que o KNN obteve resultados superiores ao LOF, mantendo a superioridade em todos os tamanhos de janelas investigados

com acurácia de 95.01% para a janela 3h, 93.91% para 6h, 91.81% para 12h e 91.54% para 24h. O que significa que em nosso trabalho, o método de detecção global com emprego de janela deslizante se revelou eficaz para a detecção de anomalia.

Na última etapa do nosso trabalho, fizemos uma avaliação dos melhores modelos empregados em cada etapa considerando outras medidas além da acurácia, como a precisão e roc. Extraímos o KNN e RF que foram as técnicas que obtiveram resultados superiores em suas etapas e ratificamos o uso da janela de 3h como o melhor tamanho de janela. Em uma aplicação de tempo real, por exemplo, há uma necessidade de uma resposta mais rápida aos eventos mantendo uma eficácia desejável do modelo. Pensando nisso, acreditamos que nossa abordagem possa ser acoplada a um sistema de detecção de alarmes, por exemplo, fornecendo uma resposta rápida ao especialista sobre o estado na colmeia.

Por fim, acreditamos que o uso desta metodologia possa contribuir na agricultura, biologia e outras áreas afins, o que reforça o caráter interdisciplinar desta pesquisa.

6.3 Trabalhos Futuros

Como trabalhos futuros, acreditamos que outros métodos não supervisionados para detecção de anomalia possam ser empregados e comparados com o intuito de potencializar o modelo melhorando a capacidade de detecção de anomalias locais. Pretendemos explorar as outras variáveis climáticas para identificar como elas podem impactar no desenvolvimento do projeto. Além disso, também acreditamos que a exploração de outros métodos para detecção em séries temporais que quando combinados com nossa abordagem possam contribuir para o alcance de melhores resultados.

6.4 Outras Contribuições

Além da dissertação apresentada, as seguintes contribuições foram obtidas no decorrer do mestrado:

- Publicação do artigo "*Improving our Understanding of the Behavior of Bees through Anomaly Detection Techniques*" por Fernando Gama, Helder M. Arruda, Hanna V. Carvalho, Paulo de Souza, e Gustavo Pessin no 26th *International Conference on Artificial Neural Networks* (ICANN), 2017. Qualis B1.

- Submissão em andamento do artigo "*A methodology for detecting anomalous events in time series from the activity level of bees*" por Fernando Gama, Helder M. Arruda, Hanna V. Carvalho, Paulo de Souza, e Gustavo Pessin para o *journal Computers*

and electronics in agriculture. Ano Submissão (2017). Qualis A2. Fator de Impacto: 2.2.

Referências

AMER, M.; GOLDSTEIN, M.; ABDENNADHER, S. Enhancing one-class support vector machines for unsupervised anomaly detection. *Proceedings of the ACM SIGKDD Workshop on Outlier Detection and Description (ODD'13)*, ACM Press, p. 8–15, 2013. Citado na página 29.

ANGIULLI, F.; PIZZUTI, C. Fast outlier detection in high dimensional spaces. *Principles of Data Mining and Knowledge Discovery*, Elomaa T, Mannila H, Toivonen H, editors, v. 2431, p. 43–78, 2002. Citado 2 vezes nas páginas 25 e 42.

BLENDER, R.; FRAEDRICH, K.; LUNKEIT, F. Identification of cyclone-track regimes in the north atlantic. *Quarterly Journal of the Royal Meteorological Society 123*, ASME Press, v. 539, p. 727–741, 1997. Citado na página 28.

BREUNIG, M. M. et al. Lof: Identifying density-based local outliers. In: . [S.l.: s.n.], 2000. Citado 4 vezes nas páginas 25, 27, 43 e 44.

CHANDOLA, V.; BANERJEE, A.; KUMAR, V. Anomaly detection : A survey. *ACM Computing Surveys*, 2009. Citado 7 vezes nas páginas 8, 17, 18, 22, 23, 27 e 28.

CHANG, C.-C.; LIN, C.-J. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, v. 2, p. 27:1–27:27, 2011. Software available at <http://www.csie.ntu.edu.tw/~cjlin/libsvm>. Citado na página 46.

CHAUDHARY, A.; SZALAY, A. S.; MOORE, A. W. Very fast outlier detection in large multidimensional data sets. *In Proceedings of ACM SIGMOD Workshop in Research Issues in Data Mining and Knowledge Discovery (DMKD)*, ACM Press, p. 78–100, 2002. Citado na página 28.

COMO, F. et al. Predicting acute contact toxicity of pesticides in honeybees apis mellifera through a k-nearest neighbor model. *Chemosphere*, 2017. Citado 2 vezes nas páginas 35 e 38.

CORTES, C.; VAPNIK, V. Support-vector network. *Machine Learning*, v. 20, p. 273–297, 1995. Citado na página 31.

EMAMIAN, V.; KAVEH, M.; TEWFIK, A. Robust clustering of acoustic emission signals using the kohonen network. *Proceedings of the IEEE International Conference of Acoustics, Speech and Signal Processing*, IEEE Computer Society, 2000. Citado na página 28.

ESKIN, E. et al. A geometric framework for unsupervised anomaly detection. *Proceedings of Applications of Data Mining in Computer Security*, Kluwer Academics, p. 78–100, 2002. Citado na página 28.

ESTER, M. et al. A density-based algorithm for discovering clusters in large spatial databases with noise. *Proceedings of Second International Conference on Knowledge Discovery and Data Mining*, E. Simoudis, J. Han, and U. Fayyad, Eds. AAAI Press, p. 226–231, 1996. Citado na página 27.

FITZGERALD, E. *California's almond farmers depend on beekeepers — and billions of bees*. 2016. Disponível em: <https://www.pri.org/stories/2016-03-20/californias-almond-farmers-depend-beekeepers-and-billions-bees>. Citado na página 17.

FOTH, M.; BLACKLER, A.; CUNNINGHAM, P. *A digital beehive could warn beekeepers when their hives are under attack*. 2016. Disponível em: <http://theconversation.com/a-digital-beehive-could-warn-beekeepers-when-their-hives-are-under-attack-54375>. Citado 2 vezes nas páginas 16 e 36.

GIANNINI, T. C. et al. Crop pollinators in brazil: a review of reported interactions. In: . [S.l.: s.n.], 2015. v. 46, p. 209–223. Citado na página 16.

GOLDSTEIN, M.; UCHIDA, S. A comparative evaluation of unsupervised anomaly detection algorithms for multivariate data. In: . [S.l.: s.n.], 2016. p. 1–31. Citado 4 vezes nas páginas 8, 25, 26 e 38.

GUHA, S.; RASTOGI, R.; SHIM, K. Rock: A robust clustering algorithm for categorical attributes. *Information Systems*, v. 25, p. 345–366, 2000. Citado na página 27.

HAQUE, S.; RAHMAN, M.; AZIZ, S. M. Sensor anomaly detection in wireless sensor networks for healthcare. In: . [S.l.: s.n.], 2015. p. 8764–8786. Citado na página 37.

HAWKINS, D. Identification of outliers. *Chapman and Hall*, 1980. Citado na página 17.

HE, Z.; XU, X.; DENG, S. Discovering cluster-based local outliers. *Pattern Recognition Letters 24*, p. 1641–1650, 2003. Citado na página 28.

HENEIN, M.; LANGWORTHY, G.; ERSKINE, J. *Vanishing of the Bees*. 2009. Disponível em: <http://www.vanishingbees.com>. Citado na página 17.

HO, T. K. Random decision forests. *Proceedings of the 3rd International Conference on Document Analysis and Recognition*, p. 2278–282, 1995. Citado na página 32.

HO, T. K. The random subspace method for constructing decision forests. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, p. 832–844, 1998. Citado na página 32.

JAGANNATH, v. *Random Forest Template for TIBCO Spotfire*. 2017. Disponível em: <https://community.tibco.com/wiki/random-forest-template-tibco-spotfirer-wiki-page>. Citado 2 vezes nas páginas 9 e 33.

JAIN, A. K.; DUBES, R. C. Algorithms for clustering data. *Prentice-Hall, Inc*, 1998. Citado na página 27.

JIANG, J. et al. A wsn-based automatic monitoring system for the foraging behavior of honey bees and environmental factors of beehives. *Computers and Electronics in Agriculture*, v. 123, p. 304–318, 2016. Citado 2 vezes nas páginas 35 e 38.

JOHNSON, J. Applied multivariate statistical analysis. *Prentice Hall*, 1992. Citado na página 17.

LEE, K. *The Importance of Pollinators*. 2010. Disponível em: <http://fruit.cfans.umn.edu/pollination/>. Citado 2 vezes nas páginas 8 e 15.

LOHNINGER, H. *Fundamentals of Statistics*. 2012. Disponível em: <http://www.statistics4u.com/fundstat_eng/cc_ann_bp_function.html>. Citado 2 vezes nas páginas 8 e 30.

LORENA, A. C.; CARVALHO, A. Uma introdução às support vector machines. 2007. Citado 3 vezes nas páginas 9, 31 e 32.

MANSHAEI, K. *Data Science*. 2015. Disponível em: <https://datascience.stackexchange.com/questions/6547/open-source-anomaly-detection-in-python>. Citado 2 vezes nas páginas 8 e 24.

MARTí, L. et al. Anomaly detection based on sensor data in petroleum industry applications. In: . [S.l.: s.n.], 2015. v. 15, p. 2774–2797. ISSN 1424-8220. Citado na página 37.

MESSAGE, D.; TEIXEIRA W.AND JONG, D. Situação da sanidade das abelhas no brasil. In: *Polinizadores no Brasil: Contribuição e Perspectivas para a Biodiversidade*. [S.l.]: Editora da Universidade de São Paulo, 2012. p. 237–356. ISBN 978-85-314-1344-5. Citado na página 16.

MURPHY, E. F. et al. b+wsn: Smart beehive with preliminary decision tree analysis for agriculture and honey bee health monitoring. *Computers and Electronics in Agriculture*, v. 124, p. 211–219, 2016. Citado 2 vezes nas páginas 35 e 38.

OZDEMIR, S.; UPADHYAYA, S. Ftda: outlier detection-based fault-tolerant data aggregation for wireless sensor networks. *Security and Communications Networks*, v. 6, p. 702–710, 2012. Citado na página 37.

POTTS S., D. et al. Global pollinator declines: trends, impacts and drivers. In: *Trends in Ecology Evolution*. [S.l.: s.n.], 2010. v. 25, p. 345–353. Citado na página 17.

RAMASWAMY, S.; RASTOGI, R.; SHIM, K. Efficient algorithms for mining outliers from large data sets. *ACM SIGMOD International Conference on Management of Data (SIGMOD'00)*, p. 427–438, 2000. Citado na página 25.

RATNIEKS F., L. W.; CARRECK N., L. Clarity on honey bee collapse? In: . [S.l.: s.n.], 2010. p. 152–153. ISSN 1095-9203. Citado na página 16.

ROSENBLATT, F. Principles of neurodynamics: Perceptrons and the theory of brain mechanisms. *Spartan Books, Washington DC*, 1961. Citado na página 30.

RUMELHART, D. E.; HINTON, G. E.; WILLIAMS, R. J. Learning internal representations by error propagation. *Parallel distributed processing: Explorations in the microstructure of cognition*, v. 1, 1986. Citado na página 30.

SARI, A. A review of anomaly detection systems in cloud networks and survey of cloud security measures in cloud storage applications. *Journal of Information Security*, v. 1, 1986. Citado 2 vezes nas páginas 8 e 22.

SCHöLKOPF, B. et al. Estimating the support of a high dimensional distribution. *Neural Computation*, v. 13, p. 1443–1471, 2001. Citado na página 29.

SHEIKHOLESLAMI, G. et al. Wavecluster: A multi-resolution clustering approach for very large spatial databases. *Proceedings of the 24rd International Conference on Very Large Data Bases*, Morgan Kaufmann Publishers Inc., p. 428–439, 1998. Citado na página 27.

SINGH, K.; UPADHYAYA, S. Outlier detection: Applications and techniques. *International Journal of Computer Science*, v. 9, n. 3, 2012. ISSN 1694-0814. Citado na página 17.

SMITH, R. et al. Clustering approaches for anomaly based intrusion detection. *Proceedings of Intelligent Engineering Systems through Artificial Neural Networks*, ASME Press, p. 579–584, 2002. Citado na página 28.

SONG, X. et al. Conditional anomaly detection. *IEEE Transactions on Knowledge and Data Engineering*, v. 19, p. 631–645, 2007. Citado na página 21.

SOUZA P., A. et al. Agent-based modelling of honey bee forager flight behaviour for swarm sensing applications. environmental modelling and software. 2017. Citado na página 16.

UPADHYAYA, S.; SINGH, K. Nearest neighbour based outlier detection techniques. *International Journal of Computer Trends and Technology*, v. 3, p. 299–303, 2012. Citado 2 vezes nas páginas 25 e 42.

VAPNIK, V. N. The nature of statistical learning theory. *Springer-Verlag New York, Inc*, 1995. Citado na página 28.

WOOLDRIGE, J. M. *Introductory Econometrics: a Modern Approach.* [S.l.: s.n.], 2000. Citado na página 21.

XIE, M. et al. Anomaly detection in wireless sensor networks: A survey. In: . [S.l.: s.n.], 2011. p. 1302–1325. Citado na página 38.

ZHANG, Y. et al. Statistics-based outlier detection for wireless sensor networks. In: . [S.l.: s.n.], 2011. v. 26, p. 1373–1392. Citado na página 36.

Apêndices

APÊNDICE A – Código fonte do trabalho

Neste apêndice nós revelamos parte dos *scripts* que foram responsáveis pelo desenvolvimento deste trabalho. Os *scripts* completos podem ser acessados pelo repositório no *Github*: <https://github.com/ffgama/swarmsensing/>.

A.1 Implementação KNN

A figura 21 apresenta o núcleo central da implementação do algoritmo KNN utilizando janela deslizante, são passados como parâmetros o tamanho da janela (*windowSize*), o número de vizinhos (*k_value*) e o limiar de decisão (*threshold*).

A figura 22 mostra os passos necessários para fazer a avaliação dos resultados do KNN utilizando matriz de confusão tomando como parâmetro os pontos sugeridos pelo usuário.

```
knn_anomaly_detection <- function(windowSize, k_value, threshold)
{

  for (row in nrow(dataset_bee):windowSize){

    window <- windowSize - 1

    # temporal interval
    # 3h - 6h - 12h - 24h
    begin <- row
    end <- begin - window

    dtset_subcj <- dataset_bee[begin:end,]

    # not remove (unlimited connections - open and close)
    closeAllConnections()

    # create a KNN
    get_scores_knn<-function(input_dataset,k){

      # to set number of instances
      n<-nrow(input_dataset)
      # create distance matrix
      distance_matrix<-as.matrix(rdist(input_dataset))

      dist_k<-NULL

      for(r in 1:n){
        # run row by row of distance matrix, sort and select farthest
        dist_k[r]<-(sort(distance_matrix[r,]))[k+1]
      }
      return(dist_k)
    }

    # to get scores
    scores<<-get_scores_knn(as.data.frame(dtset_subcj[,c("act")]), k=k_value)

    # organizing in matrix data
    scores_data <- matrix(scores, nrow = nrow(as.data.frame(dtset_subcj$act)),
                          ncol = ncol(as.data.frame(dtset_subcj$act)))
    colnames(scores_data) <- c(1:ncol(scores_data))
    rownames(scores_data) <- c(rownames(dtset_subcj))

    cat("\n==========")
    print(mean(scores_data))
    # to get average of distances (scores)
    average[row]<<-c(mean(scores_data), average)

    # threshold definition
    threshold<<-threshold

    if (any(average[row] >= threshold))
    {
      label_class <- c(label_class, "anormal")

      # filling anomaly density
      df_density_a <- rbind(average[row], df_density_a)

      # filling normal and anomaly density
      df_density_all <- rbind(average[row], df_density_all)

    }else{
      label_class <- c(label_class, "normal")

      # filling normal density
      df_density_n <- rbind(average[row], df_density_n)

      # filling normal and anomaly density
      df_density_all <- rbind(average[row], df_density_all)
    }

    df_density_a <<- df_density_a
    df_density_n <<- df_density_n
    df_density_all <<- df_density_all
    label_class <<- label_class

  }

  return(label_class)

}
```

Figura 21 – Código fonte do algoritmo KNN com janela deslizante.

```
#################################################################
############## KNN: Evaluation Metrics ##########################
#################################################################

rm(list=ls())

# setwd(paste(getwd(),"/_projeto_dissertacao/code/scripts/knn", sep=""))

load("load_data/dataset_bee.RData")
load("load_data/knn.RData")
load("load_data/test_parameters_knn.RData")

dataset_user<-read.csv("data/label_usuario.csv", header=FALSE, stringsAsFactors = FALSE)
colnames(dataset_user)<-c("classe")

# organizing suggested points algorithms and user
user<-factor(dataset_user$classe[1:length(label_class)], levels = c("anormal","normal"))
user[!complete.cases(user)]<-c("normal")
algorithm<-factor(label_class, levels = c("anormal","normal"))

# rename the factor levels
levels(user)[levels(user)=="anormal"] <- "anomaly"
levels(algorithm)[levels(algorithm)=="anormal"] <- "anomaly"

truth_table<-table(user, algorithm)

###### ######
###### ###### Results : evaluation metrics
###### ######
results<-confusionMatrix(truth_table,  mode = "prec_recall")
results

# cat("threshold", threshold)

# obter roc
roc_result<-roc(as.numeric(user), as.numeric(algorithm))
roc_result

# gravar os resultados
metrics <- data.frame(cbind(t(results$overall),t(results$byClass), roc=as.numeric(roc_result$auc)))
#write.csv(metrics,file="results_3h.csv")

save(list=ls(), file="load_data/evaluation_knn.RData")
```

Figura 22 – Código fonte responsável pela avaliação do algoritmo KNN.

A.2 Implementação LOF

A figura 23 apresenta o núcleo central da implementação do algoritmo KNN utilizando janela deslizante, são passados como parâmetros o tamanho da janela (*windowSize*), o número de vizinhos (*k_interval*) e o limiar de decisão (*threshold*).

```
lof_anomaly_detection <- function(windowSize, k_interval, threshold)
{

  for (row in nrow(dataset_bee):windowSize){

    window <- windowSize - 1

    # temporal interval
    # 3h - 6h - 12h - 24h
    begin <- row
    end <- begin - window

    dtset_subcj <- dataset_bee[begin:end,]

    # not remove (unlimited connections - open and close)
    closeAllConnections()

    # to get scores
    scores<<-lof(dtset_subcj[,c("act")], k=k_interval)

    scores_data <- matrix(scores, nrow = nrow(dtset_subcj), ncol = nrow(scores))
    colnames(scores_data) <- c(1:ncol(scores_data))
    rownames(scores_data) <- c(rownames(dtset_subcj))

    # replace by zero if there Inf or NaN.
    scores_data[which(scores_data==Inf)] <- 0
    scores_data[is.nan(scores_data)] <- 0
    print(scores_data)

    # threshold definition
    threshold<<-threshold


    if (any(scores_data[1,] >= threshold))
    {
      label_class <- c(label_class, "anormal")
      # select the maximum score
      df_density <- rbind(max(scores_data[1,which(scores_data[1,] >= threshold)]), df_density)

    }else{

      label_class <- c(label_class, "normal")
      # select the maximum score
      df_density <- rbind(max(scores_data[1,which(scores_data[1,] < threshold)]), df_density)
    }

    print(label_class)
    print(df_density)

    df_density <<- df_density
    label_class <<- label_class

  }

  return(label_class)

}
```

Figura 23 – Código fonte do algoritmo LOF com janela deslizante.

A figura 24 apresenta as avaliações dos resultados do LOF utilizando matriz de confusão tomando como parâmetro os pontos sugeridos pelo usuário.

```r
#################################################################
############## LOF: Evaluation Metrics ##########################
#################################################################

rm(list=ls())

# setwd(paste(getwd(),"/_projeto_dissertacao/code/scripts/lof", sep=""))

load("load_data/dataset_bee.RData")
load("load_data/lof.RData")
load("load_data/test_parameters_lof.RData")

dataset_user<-read.csv("data/label_usuario.csv", header=FALSE, stringsAsFactors = FALSE)
colnames(dataset_user)<-c("classe")

# organizing suggested points algorithms and user
user<-factor(dataset_user$classe[1:length(label_class)], levels = c("anormal","normal"))
user[!complete.cases(user)]<-c("normal")
algorithm<-factor(label_class, levels = c("anormal","normal"))

# rename the factor levels
levels(user)[levels(user)=="anormal"] <- "anomaly"
levels(algorithm)[levels(algorithm)=="anormal"] <- "anomaly"

truth_table<-table(user, algorithm)

###### ######
###### ###### Results : evaluation metrics
###### ######
results<-confusionMatrix(truth_table,  mode = "prec_recall")
results

# cat("threshold", threshold)

# obter roc
roc_result<-roc(as.numeric(user), as.numeric(algorithm))
roc_result

# gravar os resultados
metrics <- data.frame(cbind(t(results$overall),t(results$byClass), roc=as.numeric(roc_result$auc)))
#write.csv(metrics,file="results_3h.csv")

save(list=ls(), file="load_data/evaluation_lof.RData")
```

Figura 24 – Código fonte responsável pela avaliação do algoritmo LOF.