CENTRO UNIVERSITÁRIO DE JOÃO PESSOA - UNIPÊ
PRÓ-REITORIA ACADÊMICA - PROAC
CURSO DE BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

FELIPE ERMESON DE LIMA SILVA

APRENDIZADO DE MÁQUINA: ANÁLISE PREDITIVA DE INTERAÇÕES PROTEÍNA-PROTEÍNA NA CAVIDADE ORAL

JOÃO PESSOA – PB

2018

FELIPE ERMESON DE LIMA SILVA

APRENDIZADO DE MÁQUINA: ANÁLISE PREDITIVA DE INTERAÇÕES PROTEÍNA-PROTEÍNA NA CAVIDADE ORAL

Monografia apresentada ao Curso de Bacharelado em Ciência da Computação do Centro Universitário de João Pessoa - UNIPÊ, como pré-requisito para a obtenção do grau de Bacharel em Ciência da Computação, sob orientação do Prof. Me. Fábio Falcão de França

JOÃO PESSOA - PB

2018

S586a Silva, Felipe Ermeson de Lima.

Aprendizado de Máquina: análise preditiva de interações proteína-proteína na cavidade oral /

Felipe Ermeson de Lima Silva. - João Pessoa, 2018.

66f.

Orientador: Prof. Me. Fábio Falcão de França.

Monografia (Curso de Ciência da Computação) –

Centro Universitário de João Pessoa – UNIPÊ.

1. Cavidade Oral Humana. 2. Interações Proteína-Proteína. 3. Aprendizado de Máquina. 4. Predição. I. Título

UNIPÊ / BC CDU – 004.4:616.314

FELIPE ERMESON DE LIMA SILVA

APRENDIZADO DE MÁQUINA: ANÁLISE PREDITIVA DE INTERAÇÕES PROTEÍNA-PROTEÍNA NA CAVIDADE ORAL

Monografia apresentada ao Curso de Bacharelado em Ciência da Computação do Centro Universitário de João Pessoa - UNIPÊ, como pré-requisito para a obtenção do grau de Bacharel em Ciência da Computação, apreciada pela Banca Examinadora composta pelos seguintes membros:

Aprovada em _____/_____/2018.

BANCA EXAMINADORA

_____

Prof. Me. Fábio Falcão de França (UNIPÊ)

_____

Prof. Dra. Aline Marques de Morais (UNIPÊ)

_____

Prof. Me. Ricardo Roberto de Lima (UNIPÊ)

A Jesus Cristo, meu único e suficiente Salvador.

AGRADECIMENTOS

Não poderia concluir este trabalho sem a ajuda de várias pessoas e acima de tudo a Deus, por estar sempre me guiando e protegendo em todos os momentos da minha vida pessoal e acadêmica.

A meu orientador Prof. Ms. Fábio Falcão e amigo, a quem gostaria de expressar minha enorme gratidão pelo apoio, ajuda e tempo disponibilizado, que tornou real este trabalho.

A minha mãe Cicera que sempre esteve presente nos momentos felizes e tristes da minha vida, aos meus avós Luís e Naiza que, com muito carinho e apoio, não mediram esforços para que eu chegasse até esta etapa da minha vida, aos meus tios que sempre me ajudaram e ao meu pai Evanes que me auxiliou em momentos da minha carreira acadêmica.

A todos os meus professores que contribuíram com seus conhecimentos para minha formação no decorrer do curso.

Aos meus amigos Bruno César, Gabriel Araújo, Matheus Araújo e José William a quem tive o prazer de conviver todos esses anos, pelas conversas sobre diversos aspectos da pesquisa, e pelos ótimos momentos de descontração.

RESUMO

A cavidade oral humana contém uma grande diversidade de microrganismos comensais e hospedeiro que se relacionam mediante as interações proteína-proteína. Contudo, alterações desse microbiota podem resultar em diversas patologias como a cárie dental e doenças periodontais. A compreensão e análise dessas interações proteicas pode desenvolver fármacos e vacinas mais eficientes, e busca de alvos para câncer. Os métodos experimentais podem prever interações entre proteínas, porém apresentam algumas limitações como alto custo financeiro, baixa cobertura de interações, possuem uma alta taxa de falsos positivos e falsos negativos, dentre outras. Este trabalho apresenta uma abordagem computacional baseada em aprendizado de máquina, para analisar predições de redes de interações proteicas presentes na saliva oral humana, para tornar possível o mapeamento de possíveis doenças. A obtenção dos dados foi realizada no banco de dados públicos *BioGrid*, *SalivaTecDB* e *UniProt*, na qual foi realizada a integração das bases de dados, pré-processamento e transformação dos dados. Em seguida, foi construído e validado o modelo. O classificador construído obteve ótimos resultados com 84% de acurácia e uma área sob curva ROC de 0,919.

Palavras-Chave: Cavidade Oral Humana. Interações Proteína-Proteína. Aprendizado de Máquina. Predição.

ABSTRACT

The human oral cavity contains a large diversity of commensal and host microorganisms that are related by protein-protein interactions. However, alterations of microbiota can result in several pathologies such as dental caries and periodontal diseases. Understanding and analyzing these protein interactions can develop more efficient drugs and vaccines, and seek targets for cancer. Experimental methods can predict interactions between proteins, but they present some limitations such as high financial cost, low coverage of interactions, high false positives and false negatives, among others. This work presents a computational approach based on machine learning to analyze predictions of protein interaction networks present in human oral saliva to make possible the mapping of possible diseases. Data collection was carried out in the public database BioGrid, SalivaTecDB and UniProt, in which the integration of databases, pre-processing and transformation of the data was performed. The model was then built and validated. The constructed classifier obtained excellent results with 84% accuracy and an area under the ROC curve of 0.919.

Keywords: Human Oral Cavity. Protein-Protein Interactions. Machine Learning. Prediction.

LISTA DE ILUSTRAÇÕES

LISTA DE GRÁFICOS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

AUC – *Area Under the Curve*, Área sob Curva

BioGRID – *Biological General Repository for Interaction Datasets*, Repositório Geral Biológico para Conjuntos de Dados de Interação

CV – *Cross-Validation*, Validação Cruzada

IA – *Artificial Inteligence*, Inteligência Artificial

ID3 – *Iterative Dichotomiser*, Dicotomizador Iterativo

k-NN – *k-Nearest Neighbor*, k-vizinho mais próximo

PCA – *Principal Component Analysis*, Análise de Componentes Principais

PPI – *Protein-Protein Interaction*, Interação Proteína-Proteína

ROC – *Receiver Operating Characteristic*, Característica de Operação do Receptor

SVM – *Support Vector Machine*, Máquina de Vetores de Suporte

UniProt – *Universal Protein*, Proteína Universal

SUMÁRIO

1 INTRODUÇÃO

A sociedade está cercada de relacionamentos entre as mais diversas entidades. As redes[1] geralmente apresentam uma forma adequada para representar essas relações. O estudo das redes complexas utiliza o formalismo matemático da teoria dos grafos, que serve como base para criar e estudar modelos (NASCIMENTO, 2015). Uma definição informal para grafo é uma estrutura abstrata de um conjunto de pontos ou elementos denominados vértices, conectados a um conjunto de dependências designado de arestas (GOLDBARG, 2012).

Diversos aspectos do mundo real podem ser modelados com grafos: interações químicas, mapas rodoviários, redes neurais artificiais, árvores genealógicas, circuitos elétricos, diagramas moleculares, dentre outras. "A predição de novas interações biológicas é, na verdade, uma instância de um problema mais genérico, a predição de links em redes complexas." (NASCIMENTO, 2015, p. 14).

Compreender essas interações é de extrema importância, pois se adquiriu um caráter importante na área medicinal. As redes biológicas celulares estão sendo aplicadas para busca de alvos e ou marcadores para câncer (CHUANG *et al*., 2007), vacinas (LACOUNT *et al*., 2005) etc.

Atualmente existem várias metodologias experimentais para predição de interações entre proteínas, como: i) dois híbridos (ITO *et al*., 2001), ii) espectromia de massa de purificação por afinidade (RIGAUT *et al*., 1999), iii) cristalografia de raios-X e espectroscopia de ressonância magnética nuclear (TONG *et al*., 2001), iv) microscopia de força atômica (YANG *et al*., 2003), dentre outras. Entretanto, esses métodos são caros, onerosos e de difícil execução, além de apresentarem limitações técnicas não aplicáveis na predição de Interações Proteína-Proteína (PPI) em larga escala, quantidade de tempo, o custo associado e o mínimo de cobertura de interações proteicas por execução na rede (COELHO *et al*., 2014).

A utilização de métodos computacionais tem se mostrado uma grande alternativa de contornar as limitações citadas, uma vez que viabiliza a simulação e predição *in silico*[2] de interações até então desconhecidas, possibilitando que as análises experimentais sejam concentradas nas interações com maior probabilidade de ocorrência.

---

[1] Representação gráfica da interação entre nós por meio de vértices.

[2] Expressão utilizada para significar um método para experimentação através da simulação computacional, que modela um fenômeno natural.

Neste trabalho, será apresentada uma abordagem computacional para analisar predições de redes de interação proteína-proteína para a cavidade oral humana baseadas em interações do banco de dados público do *BioGrid*[3] (Repositório Geral Biológico para Conjuntos de Dados de Interação) junto com a base de dados *SalivaTecDB*[4].

1.1 RELEVÂNCIA DO ESTUDO

O corpo humano apresenta-se como um meio de várias comunidades de microrganismos se alojarem, se apossando de várias partes do corpo que lhe dão uma prosperidade de ciclo de vida, assegurando o seu crescimento e aumentando sua taxa de reprodução, já que tais ambientes lhes fornecem uma série de fatores como: calor, humidade e nutrientes (MURRAY, 2013 apud SOARES, 2014).

Na cavidade oral humana existe uma grande quantidade de microrganismos comensais e hospedeiro que se interagem. Para entender como esse processo biológico funciona, faz se necessário compreender como essas interações estão organizadas, tanto em sua forma estrutural como funcional. Para isso, existem vários métodos experimentais capazes de prever interações em redes biológicas. No entanto, tais métodos tem um custo financeiro e de tempo necessário para realização alto, uma baixa cobertura de rede de interações, além de possuírem uma taxa alta de falsos positivos e falsos negativos (XIA, 2010 apud SOARES, 2014).

Este trabalho proporciona uma investigação profunda sobre as principais proteínas envolvidas em infecções orais, que podem ser usadas para diagnóstico, como biomarcadores moleculares ou para tratamento, como alvos de drogas.

Assim, é evidente a necessidade de contornar tantos problemas. Um caminho promissor é realizar uma abordagem computacional baseada em aprendizado de máquina, que utiliza um conjunto de dados de PPIs obtidos por meio de métodos experimentais, para encontrar padrões em dados com a intenção de analisar uma rede mais ampla de PPIs (COELHO, 2013 apud SOARES, 2014).

[3] Disponível em: <https://thebiogrid.org/>. Acesso em: 09 jun. 2018.

[4] Disponível em: <http://salivatec.viseu.ucp.pt/salivatec-db/main.php>. Acesso em: 10 set. 2018.

1.2 OBJETIVO GERAL

Realizar uma análise computacional preditiva entre as redes de interação proteína-proteína interespécies presentes na saliva oral humana, para tornar possível o mapeamento de possíveis doenças.

1.3 OBJETIVOS ESPECÍFICOS

Como objetivos específicos, este trabalho possui:

- Construir um mapa de interações entre proteínas presentes na saliva, com base no BioGrid, SalivaTecDB e UniProt[5], com a finalidade de utilizá-lo para construção do modelo preditivo.
- Realizar o pré-processamento e transformação no conjunto de dados, com o intuito de obter apenas os dados necessários e ideais para aplicação dos algoritmos.
- Construir um modelo preditivo a fim de classificar estas interações.
- Validar o modelo criado com base em métricas obtidas através da matriz de confusão, como: acurácia, área sob curva ROC e curva de aprendizado.

1.4 INDICAÇÃO DA METODOLOGIA

Do ponto de vista de sua natureza, se apresenta uma pesquisa aplicada, pois tem como objetivo gerar conhecimento para aplicação prática, a fim de ser usado a problemas específicos (PRODANOV; FREITAS, 2013).

Neste trabalho é realizado uma abordagem indutiva, no qual parte de observações de fatos ou fenômenos em busca de suas causas, ou seja, levar a conclusões mais gerais do que suas premissas (CERVO; BERVIAN; SILVA, 2007).

Como métodos de procedimento técnicos é exercido o estatístico, que consiste em leis probabilísticas com o intuito de obter um razoável grau de precisão (GIL, 2008 apud PRODANOV; FREITAS, 2013) e também o experimental, pois define-se o objeto de estudo, e as variáveis capazes de influenciá-lo, assim como as formas de controle de maneira sistemática

[5] Disponível em: <https://www.uniprot.org/>. Acesso em: 10 nov. 2018.

(PRODANOV; FREITAS, 2013). Há uma interferência nas variáveis independentes, a fim de observar os efeitos na variável dependente (PRODANOV; FREITAS).

Quanto aos objetivos da pesquisa, tem caráter exploratório, a fim de obter novos conhecimentos e proporcionar uma melhor familiaridade com o problema (CERVO; BERVIAN; SILVA, 2007).

Do ponto de vista da abordagem do problema, a pesquisa é quantitativa, pois traduz informações em números, por meio de técnicas estatísticas, a fim de classificar e analisar esses dados (PRODANOV; FREITAS, 2013).

Os dados foram coletados no banco de dados BioGrid da versão 3.5.165 (1.583.787 interações entre proteínas, sendo 439.366 interações entre proteínas do organismo *Homo Sapiens*), SalivaTecDB (28.358 proteínas de diferentes organismos encontrados na saliva da cavidade oral humana) e UniProt (558.590 proteínas com suas respectivas sequências de aminoácidos).

Houve uma integração entre os dados desses três repositórios. A integração consiste em construir um novo *data set*, que contenha as sequências de aminoácidos contidos no UniProt para as respectivas proteínas contidas no SalivaTecDB, ainda na fase de integração, foi inserido no novo *data set* apenas pares de proteínas que continha interação no BioGrid. Assim, foi construído o conjunto de dados positivo (proteínas que se interagem).

Construiu-se ainda na fase de integração, um *script* para realizar as combinações de proteínas que possivelmente não interagem, chamado de *data set* negativo, com base nas proteínas do conjunto de dados positivo.

Foi realizado um pré-processamento para eliminação de instâncias repetidas, valores nulos e dimensões sem importância para a predição da classe, que é de caráter binário (interage ou não interage). Elaborou-se um *script* para verificação das sequências de aminoácidos, com o intuito de analisar a sanidade da sequência (se a sequência contém somente aminoácidos dos 20 tipos existentes), as sequências que continham outros caracteres não pertencentes aos 20 tipos foram eliminadas.

Após o pré-processamento, utilizou-se a biblioteca *protr*[6] da linguagem de programação R, a fim de transformar as sequências de aminoácidos em descritores. Foram utilizados os descritores Tríade Conjunta (686 *features*), Auto correlação normalizada de Moreau-Broto (480 *features*) e Composição do descritor CTD (42 *features*), totalizando 1208 *features*.

[6] Disponível em: <https://cran.r-project.org/web/packages/protr/vignettes/protr.html>. Acesso em: 10 nov. 2018.

Depois disto, se aplicou diversos algoritmos que realizam tarefas de classificação supervisionada, com a técnica de Validação Cruzada estratificada, também chamada de CV (*Cross-Validation*) com 5-*folds* (no qual consiste em separar aleatoriamente todos os dados em cinco partes, sendo um para teste e os outros quatro para treinamento, e assim sucessivamente, até todas as partes terem sido testadas) em todo o conjunto de dados, para verificar qual algoritmo obtém um melhor desempenho inicial, baseado na acurácia e área sob curva ROC. Então foi selecionado os que obtiveram os melhores resultados, para então realizar ajustes nos hiperparâmetros do algoritmo, com o intuito de aumentar o valor da acurácia e área sob curva ROC do modelo.

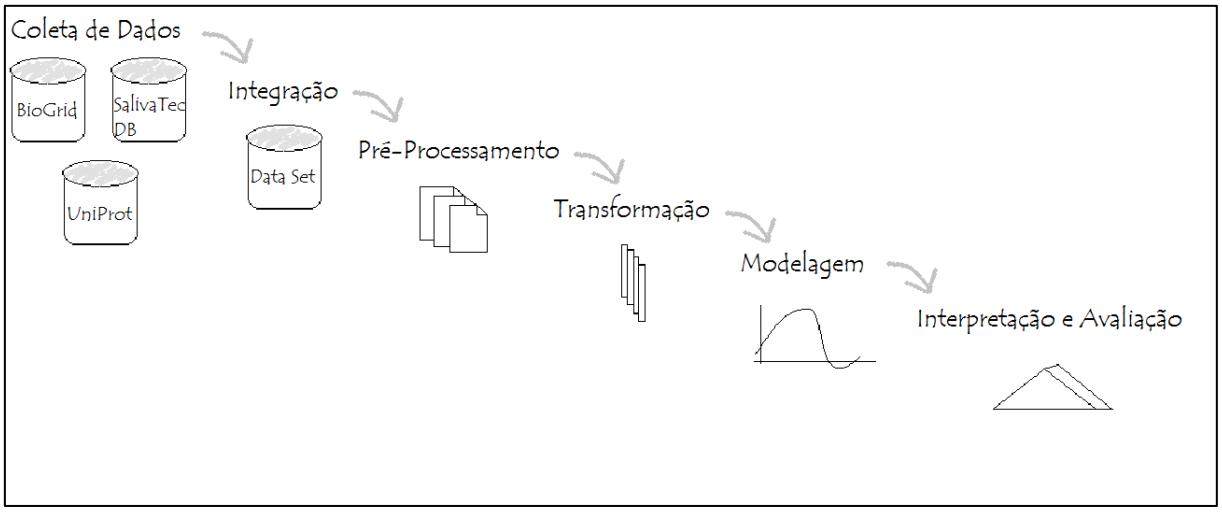
Para isso, foi divido todo o conjunto de dados em 70% para dados de treinamento, e 30% para dados de teste, e realizado os ajustes nos hiperparâmetros do algoritmo. Depois foi dividido todo o conjunto de dados em 80% para dados de treinamento e 20% para dados de teste. E feito o *tuning* necessário.

Após todo esse procedimento, realizou-se as devidas avaliações, como a acurácia, para escolha do modelo que obteve melhor desempenho.

Foi aplicado uma técnica de redução de dimensionalidade no conjunto de dados de treinamento e realizado testes no algoritmo que obteve melhor desempenho.

A Figura 1 mostra as fases de toda a metodologia usada para criação desta pesquisa.

Figura 1 – Etapas metodológicas



Fonte: Própria autoria (2018).

Vale ressaltar que, a fase de integração dos três bancos de dados, o algoritmo de combinação de possíveis proteínas que não se interagem, pré-processamento, criação e

validação do modelo, foram realizadas na plataforma *Jupyter Notebook*[7], com o auxílio da linguagem de programação *Python* 3[8], biblioteca *Pandas*[9] para análise dos dados e *Scikit-Learn*[10] para processamento de algoritmos de aprendizado de máquina. A leitura da sequência de aminoácidos contidas no repositório do UniProt foram realizadas através da biblioteca *BioPython*[11].

Os resultados foram apresentados em forma de tabelas e gráficos, com a ajuda da biblioteca *MatPlotLib*[12] e *Seaborn*[13].

O cenário em que foi desenvolvido toda a pesquisa, foi em um notebook Dell, com sistema operacional Windows 8.1, processador intel inside core i7, 1 TB de HD e memória RAM de 16 GB.

1.5 ORGANIZAÇÃO DO TRABALHO

Após esse capítulo introdutório, o conteúdo deste trabalho organiza-se da seguinte forma:

- Capítulo 2 – APRENDIZADO DE MÁQUINA E SEUS PARADIGMAS apresentará quais teorias e respectivos autores mais contribuíram para a realização do estudo e as bases teóricas para a realização deste trabalho;
- Capítulo 3 – ANÁLISE PREDITIVA DE INTERAÇÕES PROTEÍNA-PROTEÍNA apresentará o desenvolvimento da pesquisa juntamente com a solução proposta e sua aplicabilidade;
- Capítulo 4 – CONSIDERAÇÕES FINAIS apresentará de forma conclusiva, respostas aos objetivos específicos propostos pelo trabalho, apresentando também limitações desta pesquisa e trabalhos futuros;
- REFERÊNCIAS;

[7] Disponível em: <http://jupyter.org/>. Acesso em: 10 nov. 2018.

[8] Disponível em: <https://www.python.org/>. Acesso em: 10 nov. 2018.

[9] Disponível em: <https://pandas.pydata.org/>. Acesso em: 10 nov. 2018.

[10] Disponível em: <https://scikit-learn.org/stable/ >. Acesso em: 10 nov. 2018.

[11] Disponível em: <https://biopython.org/>. Acesso em: 10 nov. 2018.

[12] Disponível em: <https://matplotlib.org/>. Acesso em: 10 nov. 2018.

[13] Disponível em: <https://seaborn.pydata.org/>. Acesso em: 10 nov. 2018.

- APÊNDICES;

2 APRENDIZADO DE MÁQUINA E SEUS PARADIGMAS

Para uma melhor compreensão do campo de estudo, são expostos nas seções a seguir conceitos pertinentes ao problema proposto. Na seção 2.1 são apresentados conceitos básicos sobre proteína, sua composição, e suas interações. Em seguida, o restante deste capítulo é dedicado a explicar conceitos relacionas a Inteligência Artificial (IA), agentes inteligentes, aprendizado de máquina e o funcionamento de diversos tipos de algoritmos.

2.1 INTERAÇÕES ENTRE PROTEÍNAS

As proteínas são polímeros sintetizados pelas células, a partir de aminoácidos (VERLI *et al*., 2014). São as moléculas mais abundantes na célula (CAMPBELL *et al*., 2006 apud VIEIRA, 2013). Cada proteína possui sua função específica, como por exemplo, enzimas e hormônios, que controlam e regulam o metabolismo do corpo humano, a insulina, que é necessário para a entrada de glicose na célula, dentre outras (VIEIRA, 2013).

Proteínas são formadas pela ligação entre aminoácidos, a partir da combinação de vinte tipos de aminoácidos, constituindo uma cadeia polipeptídica. Onde a junção de um aminoácido com outro, ocorre a liberação de água (reação de desidratação). Vale ressaltar, que o corpo humano possui apenas vinte tipos de aminoácidos. Apesar de serem descritos mais de trezentos tipos de aminoácidos presentes na natureza, apenas vinte são codificados pelo material genético da célula. Dos quais nove são aminoácidos essenciais (o corpo não sintetiza, são obtidos através de alimentos) e onze aminoácidos naturais (produzidos pelo próprio organismo).

Os aminoácidos (exceto a prolina, pois possui um grupo amino secundário) são constituídos por grupos funcionais como amino primário (-NH2) e carboxila (-COOH), um átomo de C (carbono) ligado a um de H (hidrogênio) e um grupo orgânico radical, que é responsável por diferenciar cada aminoácido (CHAMPE *et al*., 2009 apud VIEIRA, 2013).

A sequência de aminoácidos é responsável pela forma de dobra e conformação tridimensional da proteína, para executar sua função bioquímica (VIEIRA, 2013). Os aminoácidos possuem variações quanto a suas propriedades físico-químicas, como aromaticidade, polaridade, basicidade, acidez, tamanho, dentre outros. E essa variabilidade torna possível uma alta gama de propriedades presentes nas proteínas. A complexidade da proteína é analisada através de quatro estruturas:

- Primária: refere-se à sequência completa de aminoácidos da proteína. É importante entender essa estrutura, pois a organização ou mutação em aminoácidos, pode provocar algumas doenças genéticas.
- Secundária: refere-se as conformações geométricas locais de uma sequência. As conformações mais estudadas são alfa-hélice e beta-plana.
- Terciária: refere-se ao empacotamento dessas estruturas secundárias, resultando em uma estrutura tridimensional.
- Quaternária: refere-se à conformação espacial da molécula, dada pela interação entre duas cadeias polipeptídicas. Só há estrutura quaternária em proteínas que possuem duas ou mais cadeias polipeptídicas.

As proteínas precisam interagir com outras moléculas (proteínas, pequenas moléculas, peptídeo) para que consigam realizar suas funções. Interações proteína-proteína são importantes em processos patológicos (REYES, 2010 apud VIEIRA, 2013).

As regiões onde ocorre a interação entre proteínas (região da superfície, onde ocorre uma interação física) é de extrema importância para caracterizar sua função e mecanismo de interação. Ainda não existem estudos que conseguem evidenciar uma propriedade capaz de identificar uma região de interface corretamente. Por isso, utiliza-se uma combinação de várias propriedades físico-químicas, a fim de descrever propriedades que definem a região de interface proteína-proteína (HIGA, 2009 apud VIEIRA, 2013).

A predição de interações entre proteínas, constitui um problema de classificação binária, com o intuito de identificar se uma determinada proteína interage com outra determina proteína ou não. Compreender essas interações, possibilita uma investigação profunda das redes de interação proteína-proteína, fornece mais informações para projeto de drogas, visando interações de uma proteína específica.

2.2 INTELIGÊNCIA ARTIFICIAL

O homem tenta ao longo do tempo compreender como um mero punhado de massa consegue raciocinar, perceber e manipular um mundo tão complexo. A inteligência artificial busca não somente entender como funciona a inteligência, mas também como construir um agente inteligente. No entanto, a IA sofre com a realidade de que a inteligência ainda é mal definida, dificultando uma precisa definição da IA e uma melhor avaliação do agente aparentemente inteligente.

Existem várias definições de IA, dentre elas a de Winston (1992 apud RUSSEL; NORVIG, 2013) é o estudo das técnicas computacionais que tornam possíveis perceber, pensar e agir. Alan Turing foi uma das maiores personalidades na história da IA. Turing publicou um artigo na revista filosófica Mind intitulado "*Computing Machinery and Intelligence*" (TURING, 1950), em que abordava a questão se era possível fazer a máquina pensar ou não. Diante das ambiguidades da própria questão de inteligência e máquina, propôs um teste para medir se o computador continha inteligência, conhecido como teste de Turing.

O teste de Turing ou jogo da imitação, consiste em uma pessoa (interrogador) dentro de uma sala e uma outra sala com uma pessoa e a máquina supostamente inteligente. O interrogador irá realizar perguntas através do seu computador e irá receber respostas da outra pessoa com um computador e da máquina supostamente inteligente. O fato de serem em salas diferentes assegura o teste de que o interrogador não será influenciado pela aparência ou por uma voz mecânica. Por fim, se o interrogador não conseguir distinguir se quem está falando é um computador ou uma pessoa, conclui-se que a máquina é inteligente.

O teste proporciona algumas vantagens como dar uma noção objetiva de inteligência evitando possíveis desvios por questões atualmente irrespondíveis como, por exemplo, se as ações feitas pela máquina foram conscientes ou não (LUGER, 2013). Em razão dessas vantagens, formou-se uma base para a avaliação de programas de IA modernos, surgindo uma variação do teste de Turing, como, por exemplo, profissionais especialistas para avaliar o desempenho do sistema especialista projetado para resolução de um conjunto de problemas complexos.

Apesar de sua importância na IA, o teste de Turing sofre críticas justificáveis. Uma delas é o fato de não testar as habilidades necessárias para a percepção. Outra crítica é que esse teste limita a capacidade de inteligência da máquina ao tentar moldar-se a mente humana (LUGER, 2013).

Alan Turing citou várias objeções, dentre elas a Objeção de Lady Lovelace formulada por Ada Lovelace, consistia em uma argumentação de que máquinas não podem ser inteligentes porque só podem realizar aquilo que foi originalmente programado. No entanto, essa refutação está se tornando dúbia, já que alguns sistemas especialistas mostraram resultados não previstos pelos seus projetistas, além de vários pesquisadores considerarem que a criatividade pode ser codificada em uma máquina.

Outra objeção citada foi o Argumento da Informalidade do Comportamento, que discute a impossibilidade de projetar um conjunto de regras que possam fazer a máquina agir a qualquer situação que venha acontecer (LUGER, 2013), ou seja, o comportamento humano é complexo demais ao ponto de não ser possível traduzi-lo em regras. Pesquisadores vem trabalhando ao longo dos anos para contornar essa deficiência, construindo modelos como as redes neurais e sistemas de raciocínio probabilístico. É fato que a flexibilidade de uma inteligência biológica em responder um conjunto infinito de situações de forma aceitável é uma componente de extrema importância para caracterização da inteligência (LUGER, 2013). Para Simon (1981, apud LUGER, 2013) essa habilidade da inteligência biológica se dá a grande variedade do ambiente em que estão situados e não ao código interno. Observando as formigas, e argumentando que a complexidade do comportamento das formigas se dá através do mundo em que vivem, se comprovado além dos insetos também para humanos, serão descobertos grandes avanços científicos.

2.3 IA FORTE E IA FRACA

Por causa de vários pesquisadores pensarem de forma distinta sobre a IA, deu início a duas perspectivas, conhecidas como IA-Fraca e IA-Forte. Adeptos que defendem a IA-Forte argumentam que máquinas tendo posse de uma capacidade de armazenamento e processamento suficiente e um código que possa proporcionar inteligência, pode-se afirmar que o computador tem consciência equivalente ao ser humano (COPPIN, 2010).

Se por um lado há os mais otimistas, do outro lado ficam aqueles mais contidos e realistas. Defensores da IA-Fraca alegam que é possível apenas implementar códigos que moldam o comportamento humano a fim de resolver problemas complexos do mundo real, afirmando que o fato de resolverem problemas complexos de forma inteligente não caracteriza que estão tendo consciência de seus atos (COPPIN, 2010).

Como uma forma de refutar a ideia da IA forte, John Searle criou o argumento da Sala Chinesa, que consiste em uma pessoa que só fala inglês dentro de uma sala fechada apenas com uma abertura para *inputs* e *outputs*. Na sala contém um conjunto de instruções em inglês e cartas com símbolos chineses. Uma história e perguntas em chinês são entregues ao ser humano que está dentro da sala. Ele lê as instruções e consegue responder em chinês as perguntas sobre a

história e as devolve. Os chineses que estão fora da sala claramente irão pensar que a sala entende chinês.

John Searle compreende que o homem, as cartas e a sala não compreende chinês, embora a máquina possa enganar chineses do lado de fora. A objeção de John Searle confronta o ponto de vista de Turing, de que se a máquina consegue enganar um ser humano, ela já pode ser considerada inteligente (COPPIN, 2010).

2.4 AGENTE INTELIGENTE E SEU AMBIENTE

Um agente é tudo que pode perceber seu ambiente por meio de sensores e realizar alguma ação no seu ambiente por intermédio de atuadores (RUSSELL; NORVIG, 2013). Os agentes podem ser biológicos, robóticos e computacionais. O ser humano é um agente inteligente biológico, e tem como sensores olhos, ouvidos, nariz, dentre outras partes. E como atuadores as pernas, mãos etc. Os robóticos podem ter diversos sensores como de temperatura, presença, movimento e como atuadores vários motores. Os computacionais ou agentes de software também apresenta sensores como por exemplo, um conjunto de entradas por teclado, arquivos, pacotes de rede e como atuadores pode escrever alguma resposta na tela, escrever em arquivos etc.

Há várias propriedades que os agentes podem ou não apresentar. Benevolência, aprendizado, colaboração, inteligência, autonomia, versatilidade e mobilidade são algumas delas. Para que um agente tenha autonomia é preciso que ao receber novas percepções antes não previstas, seja capaz de realizar uma ação ao ambiente, com base em suas antigas percepções. Os agentes podem interagir entre si para resolver um problema complexo em conjunto, demonstrando assim sua benevolência ou colaboração. Os agentes também podem ter a capacidade de aprender tanto individualmente como em conjunto em sistemas multiagentes. No aprendizado, é possível através da experiência melhorar sua performance, evitando erros iguais no futuro na resolução de problemas (COPPIN, 2010).

Os agentes são classificados em diversos tipos, são esses: agentes reativos, baseados em objetivos, baseados em utilidade, móveis, colaborativos etc. É importante ressaltar que os tipos de agentes não são mutuamente excludentes. Agentes reativos escolhem ações com base nas regras predefinidas para a percepção atual, independentemente de seu histórico de percepções

e não são ideais para ambientes em constante mudança. Agentes baseados em objetivos e utilidade são semelhantes pois ambos buscam atingir um objetivo. Porém, os baseados em utilidade se preocupam em oferecer resultados eficazes, enquanto os baseados em objetivos dão importância apenas em reconhecer se o objetivo foi alcançado ou não.

Há numerosas categorias de ambientes quando se trata de IA. Ambiente observável, parcialmente observável, inobservável, determinístico, estocástico, episódico, sequencial, estático, dinâmico, conhecido, desconhecido, e muitos outros. Ao se projetar um agente é de extrema importância definir o ambiente em que o agente irá atuar, para definir técnicas de implementação e o projeto apropriado (RUSSELL; NORVIG, 2013).

Enquanto IA pode ser definida como uma ciência de modo ampla, capaz de imitar diversas faculdades humanas, aprendizado de máquina é uma vertente da IA, que treina máquinas para construir modelos computacionais, capazes de aprender através da experiência (dados).

2.5 APRENDIZADO DE MÁQUINA

Não há como pensar em uma máquina inteligente sem possuir a capacidade de aprender. O aprendizado de máquina possui várias aplicações para problemas reais como: aprender a configuração de cada novo labirinto, prever os valores do mercado de ações amanhã, diagnosticar câncer por meio de análise de dados de expressão gênica[14], dentre outras. O aprendizado de máquina pode ser entendido como a faculdade de aperfeiçoar suas tarefas com base no seu conhecimento adquirido anteriormente (MITCHELL, 1997 apud FACELI *et al*., 2011).

O processo de aprendizado, geralmente acontece no momento do treinamento. A máquina tenta aprender de acordo com um conjunto particular de dados fornecido, classificando esses mesmos dados durante o treinamento e também dados não observados durante o teste.

Para conseguir a façanha de aprendizado, é posto um princípio de inferência chamado indução, onde se pretende obter generalizações com base em um conjunto de dados. A linha de raciocínio desse princípio é que se houver uma função, hipótese ou regra que possa classificar

[14] Processo em que a informação contida em um gene é traduzida em mRNA ou proteína.

corretamente os dados do treinamento, é muito provável que também consiga classificar com precisão os dados que ainda não foram percebidos.

O conjunto de hipóteses que mapeie corretamente o conjunto de dados de treinamento tanto positivos como negativos, é chamado de espaço de versão. Uma forma de aprender é por meio do espaço de versão, eliminando todas as hipóteses que não consigam classificar corretamente, porém em um ambiente complexo, o fato de especificar cada hipótese é inviável (COPPIN, 2010).

No momento do treinamento, o algoritmo está buscando uma hipótese no espaço de possíveis hipóteses que relacione corretamente os objetos e um melhor ajuste com o conjunto de dados de treinamento (FACELI *et al*., 2011). Uma vez induzida a hipótese, há uma forma de representá-la. As redes neurais artificiais, por exemplo, utilizam um conjunto de valores reais com seus respectivos pesos, a árvore de decisão é representada através de uma árvore com perguntas nos seus nós, dentre outras. Há um outro viés, chamado de viés de busca, que define como será encontrado a hipótese no espaço de hipóteses possíveis. Essa predisposição indutiva é encontrada em todos os métodos de aprendizado (COPPIN, 2010).

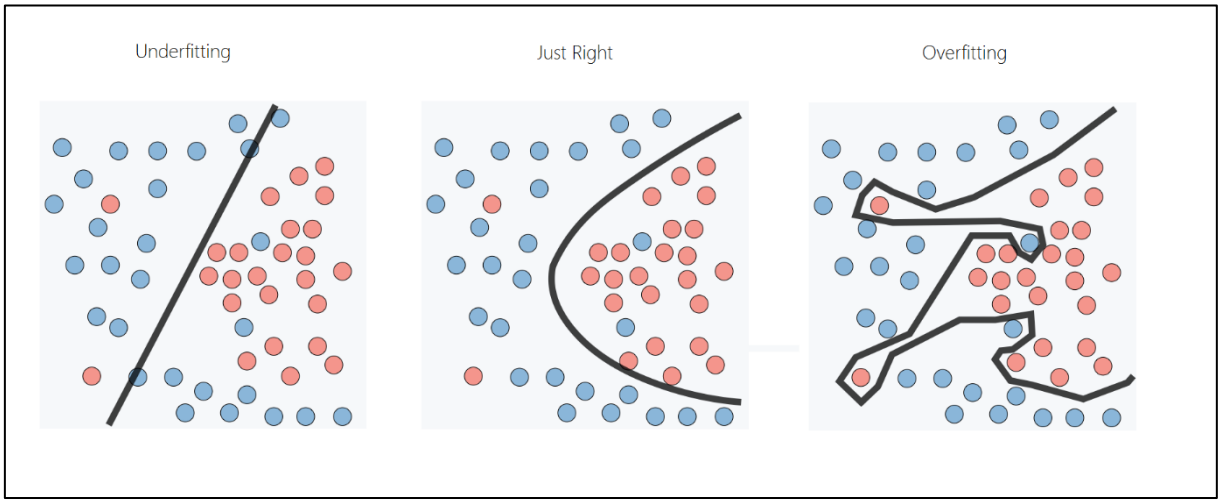
É de grande valia, que os métodos de aprendizagem contenham essas restrições para limitar as hipóteses a serem visitadas no espaço de busca, pois caso contrário, acarretaria em um espaço de versão com todas as hipóteses possíveis impondo que a hipótese encontrada pelo algoritmo nunca seria geral o suficiente para classificar corretamente dados até então não observados.

O conjunto de dados de treinamento por vezes possuem inconsistências, ruídos, classe rara, dados ausentes ou redundantes. Isso acaba comprometendo o processo indutivo, e uma forma de diminuir essa influência é realizando um pré-processamento.

Quando um modelo acaba perdendo a capacidade de generalização, e consegue apenas classificar os dados de treinamento, afirma-se que ouve um superajuste aos dados de treinamento ou *overfitting*. Na ocasião de acontecer o processo inverso, mostrar baixas taxas de acertos no treinamento, devido ao modelo ser simples, ou os dados de treinamento é pouco representativo ou não consegue obter padrões nos dados, denomina-se que ocorreu uma situação de *underfitting*. No momento em que ocorre bons resultados, tanto nos testes como em produção do modelo construído, conclui-se que o modelo é genérico, aconteceu uma generalização (AMARAL, 2016).

A Figura 2 mostra um exemplo de *underfitting*, no centro um exemplo de modelo ideal, e a direita um exemplo de *overfitting*.

Figura 2 – Exemplos de classificadores com *underfitting*, *just right* e *overfitting*.



Fonte: Stanford ([20--])[15]

A maldição da dimensionalidade também é um problema de superajuste. Se caracteriza como um conjunto de dados que possui muitas dimensões (atributos ou colunas), acarretando o mau funcionamento do modelo. Nem sempre ter muitas dimensões é sinônimo de um bom desempenho. A redução de dimensões pode melhorar algoritmos de aprendizagem, melhorar o custo de processamento de grandes conjuntos de dados além de facilitar uma melhor compreensão dos dados analisados. Uma forma de reverter essa maldição da dimensionalidade, é compreender bem o negócio, identificando atributos que não possuem importância para a construção do modelo (AMARAL, 2016).

Contudo, nem sempre é possível identificar os atributos que possuem ou não relevância. Por esse motivo, há várias técnicas de seleção de atributos, que são divididas em três categorias: *wrapper*, *embeded* e filtro (LIMA, 2016).

Outro problema de superajuste é a classe rara. Ao analisar o conjunto de dados de treinamento, talvez não tenha dados suficientes para classificar uma determinada classe, provocando a não generalização do modelo (AMARAL, 2016). Uma possível solução é realizar

[15] Disponível em: <https://stanford.edu/~shervine/l/pt/teaching/cs-229/dicas-truques-aprendizado-maquina#diagnostics>. Acesso em: 03 nov. 2018.

uma estratificação, no qual consiste em dividir proporcionalmente para que se consiga uma generalização e o modelo possa também compreender corretamente essas classes raras.

Para realizar a construção do modelo, é preciso dividir o conjunto de dados (dados históricos) em dois subconjuntos. Um para realizar o treinamento do modelo, e o outro para efetuar o teste (validação) do modelo. Há várias formas de executar a divisão dos dados históricos.

A mais comum é chamada de *hold out*, que consiste em dividir aleatoriamente 70% para treinamento e 30% para teste os dados históricos sem substituição, ou seja, os dados que foram selecionados para o treinamento não poderão ser selecionados para o subconjunto de teste (AMARAL, 2016). Os dados serão selecionados de modo que as chances sejam iguais para qualquer instância.

Quando há uma divisão em dois conjuntos (treinamento e teste) pode ser que exista um exemplo que melhor generalize a tarefa, o qual pode estar contido no conjunto de teste, o que faz ser um problema, pois aquele exemplo poderia estar no conjunto de treinamento e não no conjunto de teste. Outro problema é quando o conjunto de dados possui poucos exemplos.

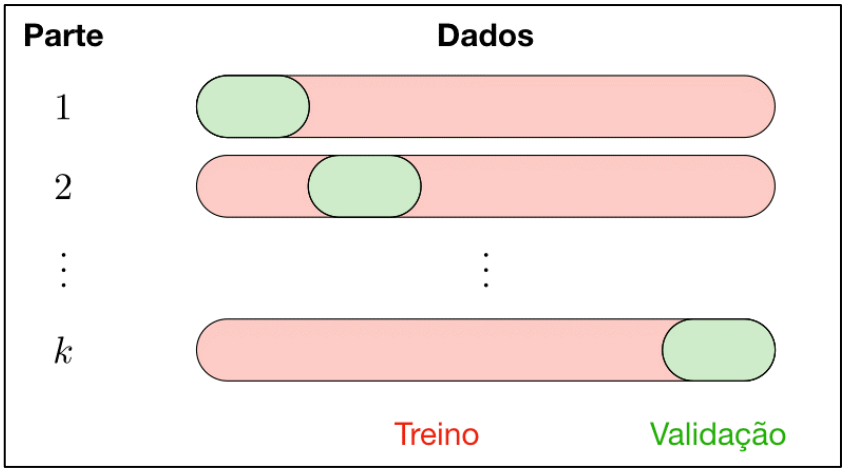
Uma alternativa para contornar esse problema é a utilização de uma técnica chamada Validação Cruzada (CV do inglês, *Cross-Validation*). A validação cruzada consiste em dividir todo o conjunto de dados em *k folds* ou partes. Onde são utilizados *k*-1 partes para treinamento e a parte que sobrou para teste. O processo é repetido *k* vezes, utilizando a cada iteração uma partição diferente para teste. Em seguida realiza-se a média da métrica escolhida de todos os testes, a fim de obter o desempenho do modelo. A Figura 3 ilustra o funcionamento da validação cruzada com 5 *folds.*

A validação cruzada estratificada é uma outra vertente da validação cruzada, onde consiste em manter na partição a proporção de exemplos de cada classe semelhante ao conjunto de dados original. Por exemplo, se o conjunto de dados inteiro tiver 60% de exemplos da classe A e 40% com exemplos da classe B, cada partição procura manter essa proporção.

Outra técnica é a *leave-one-out*, no qual o k = n, onde n representa a quantidade de exemplos do conjunto de dados. Esse método vai realizar o treinamento com n – 1 exemplos, e testar apenas um exemplo, o processo se repete n vezes.

Essa técnica produz uma estimativa mais fiel do desempenho do modelo, porém é computacionalmente caro, é indicado para conjunto de dados pequenos (FACELLI et al., 2011).

Figura 3 – Validação Cruzada.



Fonte: Stanford ([20--])[16].

Feito a classificação do conjunto de teste, é preciso testar o quanto foi eficiente o modelo. Para isso, faz-se necessário comparar os dados de teste com os dados previstos. Isso irá gerar uma matriz de confusão, que consiste em uma tabela com os registros de verdadeiros positivos (VP) que são instâncias que o modelo classificou como positivo e de fato era positivo, verdadeiros negativos (VN), que são instâncias que eram falsas e de fato era falso, falso positivo (FP) que são instâncias que foram classificadas como positivo e na verdade era falso e falso negativo (FN) que foram classificados como dados negativos e que na verdade são positivos. A Tabela 1 apresenta a matriz de confusão.

Tabela 1 – Matriz de confusão.

	PREDIÇÃO	
DADOS REAIS	Verdadeiros Positivos	Verdadeiros Negativos
Classif. como positivos	VP	FP
Classif. como negativos	FN	VN

Fonte: Própria autoria (2018).

Pode-se obter vários indicadores ou medidas de desempenho a partir da matriz de confusão, como:

[16] Disponível em: <https://stanford.edu/~shervine/l/pt/teaching/cs-229/dicas-truques-aprendizado-maquina#model-selection>. Acesso em: 03 nov. 2018.

Ademais, n = VP + VN + FP + FN.

- *Taxa de erro na classe positiva*: proporção entre os exemplos positivos classificados incorretamente, também conhecido como taxa de falsos negativos (TFN).

$$err_+ = \frac{FN}{VP + FN} \tag{1}$$

- *Taxa de erro na classe negativa*: proporção entre os exemplos negativos classificados incorretamente, também conhecido como taxa de falso positivos (TFP).

$$err_- = \frac{FP}{FP + VN} \tag{2}$$

- *Taxa de erro total*: proporção entre a soma dos valores da diagonal secundária da matriz e a soma de todos os valores da matriz.

$$err = \frac{FP + FN}{n} \tag{3}$$

- *Taxa de acerto ou acurácia total*: proporção entre a soma dos valores da diagonal primária e a soma de todos os valores da matriz.

$$ac = \frac{VP + VN}{n} \tag{4}$$

- *Precisão*: proporção entre os exemplos positivos classificados corretamente e todos os preditos como positivos.

$$prec = \frac{VP}{VP + FP} \tag{5}$$

- *Sensibilidade*: corresponde à taxa de acerto na classe positiva. Conhecida como taxa de verdadeiros positivos (TVP).

$$sens = \frac{VP}{VP + FN} \tag{6}$$

- *Especificidade*: corresponde à taxa de acerto na classe negativa. Seu complemento corresponde à taxa TFP.

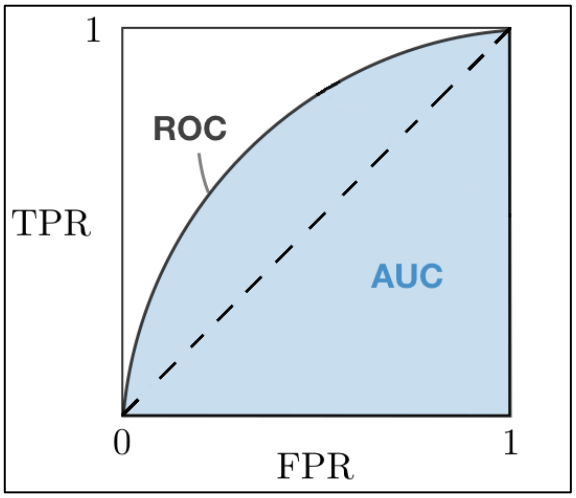
$$esp = \frac{VN}{VN + FP} = 1 - TFP \tag{7}$$

Outra forma de avaliar classificadores em problemas binários, é por meio das curvas ROC (*Receiving Operating Characteristics*) (FAWCETT, 2005 apud FACELLI et al., 2011).

O gráfico ROC é um gráfico bidimensional, onde o eixo x representa a taxa de falsos positivos (TFP) e o eixo y representa a taxa de verdadeiros positivos (TVP). Para comparar classificadores com base na curva ROC, e por muitas das vezes as curvas se interceptam, dificultando a análise, é preciso reduzir a curva a um escalar. Utiliza-se a área sob a curva

(AUC, do inglês *Area Under the Curve*) como forma de avaliação. A Figura 4 ilustra uma curva ROC e sua AUC. Onde TPR representa a Taxa de Verdadeiros Positivos e FPR significa a Taxa de Falsos Positivos.

Figura 4 – ROC e AUC



Fonte: Stanford ([20--])[17].

A linha diagonal tracejada representa a aleatoriedade, modelos que ficam abaixo dessa linha são piores que uma função que classifica de forma aleatória. Ao comparar classificadores, não havendo intersecção das curvas, a curva que melhor se aproxima da coordenada (0,1) possui um melhor desempenho. No caso de haver intersecções, ou seja, regiões que um é melhor do que o outro, e vice-versa, a melhor forma de avaliar é utilizando a área sob curva ROC.

Vale ressaltar que o aprendizado de máquina possui seus limites. O fato de quase tudo ser *datificado*[18] deu um poder enorme a IA. No entanto, se uma determinada pessoa é classificada como um bom pagador do empréstimo, e tem um filho prestes a se casar e o casamento causou um endividamento para o sujeito, a probabilidade de não pagar o empréstimo é enorme. Pode-se melhorar a precisão do modelo por meio de várias estratégias, mas se essa informação não estiver no conjunto de dados, irá falhar. Dito isto, o aprendizado é totalmente refém da informação.

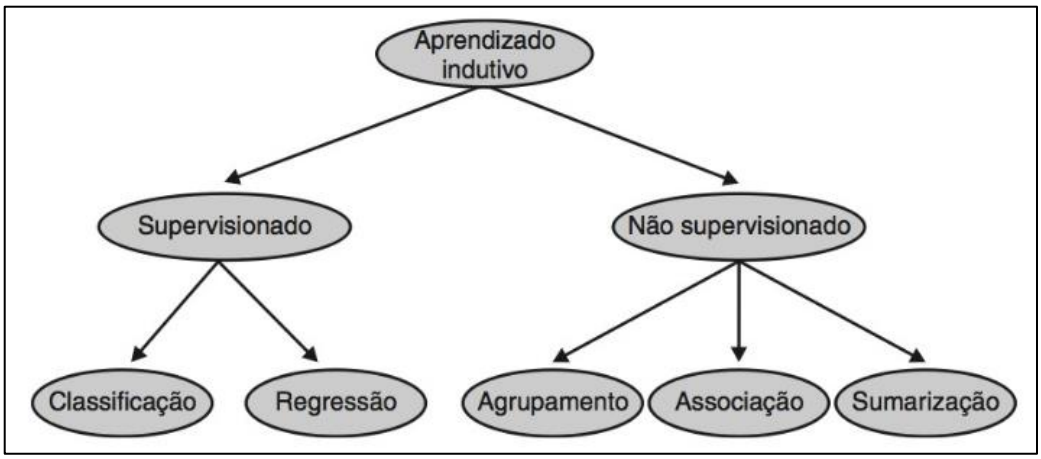
No aprendizado de máquina, divide-se o aprendizado indutivo em dois grandes paradigmas: o aprendizado supervisionado (*supervised learning*) e o aprendizado não

[17] Disponível em: <https://stanford.edu/~shervine/l/pt/teaching/cs-229/dicas-truques-aprendizado-maquina#classification-metrics>. Acesso em: 03 nov. 2018.

[18] Expressão usada para indicar o processo de transformar qualquer coisa em dado digital.

supervisionado (*unsupervised learning*). A Figura 5 mostra que na abordagem supervisionada existem dois tipos de tarefas (classificação e regressão) e na outra abordagem temos três tipos de tarefas (agrupamento, associação e sumarização).

Figura 5 - Classificação das abordagens do aprendizado de máquina.



Fonte: Faceli *et al*. (2011)

2.6 ANÁLISE DE COMPONENTES PRINCIPAIS (PCA)

Muitos problemas reais que podem ser resolvidos por meio do aprendizado de máquina, apresentam uma grande quantidade de atributos, em seu conjunto de dados. Um exemplo de problemas com uma elevada quantidade de atributos, são os dados da expressão gênica, quando se está trabalhando com objetos como amostra e genes como atributos (FACELI *et al*., 2011).

Grandes quantidades de atributos podem ser prejudiciais ao aprendizado, tendendo ao mal funcionamento do modelo (AMARAL, 2016). Esse problema é descrito como maldição da dimensionalidade (FACELI *et al*., 2011).

A redução da dimensionalidade proporciona um melhor desempenho do modelo induzido, o custo computacional diminui e melhora a compreensão dos resultados obtidos (FACELI *et al*., 2011).

Existe duas abordagens para evitar o problema da dimensionalidade: i) agrupamento, onde realiza a combinação de atributos, formando um novo atributo; e a ii) seleção de atributos, na qual mantém parte dos atributos, mas os atributos irrelevantes ou redundantes são eliminados.

Uma das técnicas mais conhecidas de agregação é a Análise de Componentes Principais (PCA, do inglês *Principal Component Analysis*). PCA é uma técnica estatística, que busca diminuir a quantidade de atributos do conjunto de dados original, preservando a mesma quantidade de informação (SANTOS, 2016).

2.7 APRENDIZADO SUPERVISIONADO

No aprendizado supervisionado, o sistema aprende com um conjunto de dados de treinamento pré-classificados. O termo supervisionado remete a um supervisor de fora que sabe a classificação adequada para cada elemento (FACELI *et al*., 2011). Suponha que o conjunto de dados é em forma de tabela, cada coluna é chamada de dimensão e cada tupla ou linha de instância. Geralmente, a classe ou variável de interesse fica na última dimensão. Ela representa a classificação da instância ou valor que se quer prever.

Há vários paradigmas na aprendizagem supervisionada. O paradigma simbólico procura aprender com exemplos positivos e negativos por meio de uma construção de representações simbólicas de um conceito. As representações simbólicas mais comuns são as árvores de decisão e as regras de decisão. Uma árvore de decisão indutiva consiste na utilização de dados históricos para produzir uma árvore de decisão que classifique corretamente novas percepções (COPPIN, 2010). O algoritmo de árvore de decisão indutiva ID3 (*Iterative Dichotomiser* 3), desenvolvido por Quinlan nos anos 1980, teve uma grande contribuição para a IA (COPPIN, 2010). O ID3 produz uma árvore a partir do topo. Cada nó possui uma pergunta com base nas características do conjunto de dados. A ordem das perguntas é de grande relevância. Há possibilidade de a escolha ser aleatória, porém não seria tão eficiente. O algoritmo ID3 escolhe as quais características usar com base no ganho de informação.

O ganho de informação por definição é redução em entropia, no qual a entropia de um conjunto de dados de treinamento, S, é definida por:

$$\mathrm{H}(S) = -p_1 \log_2 p_1 - p_0 \log_2 p_0 \qquad (8)$$

onde p_1 é a proporção de dados positivos e p_0 a proporção de dados negativos. A entropia de S será seu valor máximo, de 1, quando exatamente metade dos dados forem positivos e a outra metade negativo. E alcançará 0 quando todos os dados forem positivos ou negativos.

O algoritmo ID3 possui uma predisposição indutiva que se expressa produzindo a menor árvore de decisão para classificação correta dos dados de treinamento (COPPIN, 2010).

Outro paradigma do aprendizado supervisionado, é o estatístico. A ideia central é encontrar uma aproximação do conceito induzido através de modelos estatísticos. Esses modelos podem ser paramétricos, quando realizam alguma inferência sobre a distribuição dos dados, ou não paramétricos, quando não realizam. Dentre os métodos estatísticos, destaca-se os métodos Bayesianos, no qual, utilizam modelos probabilísticos com um conhecimento prévio, combinado com instâncias do conjunto de dados de treinamento, a fim de determinar a probabilidade de uma hipótese.

Uma forma de classificação é "lembrar" de casos similares cuja classe é conhecida e classificar a nova instância de acordo com o caso lembrado. Esse é um paradigma conhecido como baseado em instância. Métodos de aprendizado em instância não realiza generalizações em busca de uma hipótese a partir do conjunto de dados de treinamento, mas armazenam todos os dados de treinamento e busca classificar novas instâncias (COPPIN, 2010).

Pelo fato de memorizar todo o conjunto de dados de treinamento, dependendo da quantidade de dados, pode ficar lento e difícil manusear. Para reverter isso, pode-se memorizar apenas casos importantes que resuma toda a informação. Outra forma de combater esse problema é construir estruturas capazes de indexar instâncias e realizar buscas eficientes. Algumas dessas estruturas são as *M-trees* e *Slim-trees*.

Quando as dimensões são quantitativas, se pode calcular a distância entre as instâncias pela computação da distância Euclidiana, e assim chegar a uma classificação. Porém quando são dimensões qualitativas, esse processo torna-se muito problemático. Existem duas formas para se classificar uma nova instância, uma delas é classificar a nova instância com base na instância mais próxima dela. E a segunda alternativa, é com base em várias instâncias de referência ao invés de apenas uma. A segunda opção é a mais robusta, e está sujeita a menos erros. Um algoritmo de aprendizado baseado em instância é o do vizinho mais próximo, e ao contrário de árvore de decisão, possui um melhor desempenho com relação a ruídos (COPPIN, 2010).

Outro paradigma é o conexionista, e um exemplo é a rede neural artificial, que consiste em uma rede de nós de processamento, inspirado no modelo biológico do cérebro humano (COPPIN, 2010). O nó da rede é análogo ao neurônio do cérebro humano. Essa rede artificial é formada por várias camadas, que podem ser divididas em três: entradas, intermediárias e

saída. A camada de entrada recebe os dados a serem classificados e isso acaba ativando outros nós da rede até que chegue na camada de saída, formando um padrão complexo de ativações.

As próximas seções apresentam diversos algoritmos que fazem parte do aprendizado supervisionado.

2.7.1 Naive Bayes

Uma forma de lidar com tarefas preditivas em aprendizado de máquina, no qual a representação de conhecimento de domínio há incertezas, é por meio de modelos probabilísticos.

Para uma melhor compreensão dessa seção, é descrito algumas noções sobre probabilidade e probabilidade condicional.

A probabilidade de um evento (*a priori*) é a razão do número do conjunto de elementos de um evento (E) e o número de elementos do conjunto de todas as possibilidades de resultado, onde denomina-se de espaço amostral (Ω) (LUGER, 2013). Assim,

$$P(E) = \frac{|E|}{|\Omega|} \tag{9}$$

Probabilidade condicionada pode ser definida como:

$$P(A|B) = \frac{P(A \cap B)}{P(B)} \tag{10}$$

onde: P(A|B) é a probabilidade (*a posteriori*) de A, dado B. A probabilidade de A está condicionada a B, ou seja, estabelece a probabilidade de A ser verdadeiro dado que B é verdadeiro. P(A∩B) é a probabilidade de que tanto A quanto B sejam verdadeiros. P(B) não pode ser zero.

O teorema de Bayes mostra que a probabilidade condicional de um evento A pode ser calculada pela probabilidade condicional inversa e pelas suas probabilidades *a priori* de A e B. Assim, a equação de Bayes é dado por

$$P(A|B) = P(A)\,\frac{P(B|A)}{P(B)} \tag{11}$$

onde: P(A|B) é a probabilidade *a posteriori* de A, dado B. P(A) é a probabilidade *a priori* de A. P(B) é a probabilidade *a priori* de B. P(B|A) é a probabilidade condicional inversa.

Para elucidar a aplicação do teorema de Bayes, é descrito um dos exemplos clássicos na literatura, que relaciona meningite com torcicolo.

Seja T o evento de um paciente ter torcicolo e M o evento de um paciente ter meningite. Considere também que a probabilidade incondicional de um paciente ter torcicolo é de P(T) = $\frac{1}{20}$ e a probabilidade incondicional de um paciente ter meningite é de P(M) = $\frac{1}{50000}$. Pelo fato de um consultório médico conter dados sobre pacientes enfermos, é possível verificar a incidência dos sintomas presentes. Assim, supõe-se que a probabilidade condicional de um paciente com meningite apresentar torcicolo, seja de P(T|M) = $\frac{1}{2}$. Para calcular a probabilidade a posteriori de M dado que o paciente tenha T, utiliza-se o teorema de Bayes. Assim,

$$P(M|T) = P(T|M)\frac{P(M)}{P(T)} \tag{12}$$

substituindo pelos devidos valores, obtêm-se

$$P(M|T) = 0{,}5\ \frac{\frac{1}{50000}}{\frac{1}{20}} \tag{13}$$

imediatamente é fácil perceber que a probabilidade a posteriori de M, dado T é de 0,02%.

Um modelo simples de aprendizado baseado no teorema de Bayes, é o agente dado um conjunto de evidências escolher em qual hipótese vai acreditar, ao reconhecer a probabilidade a posteriori de cada uma das hipóteses (COPPIN, 2010).

Dado um conjunto de possíveis hipóteses, $H_{1, \ldots ,}H_{n,}$ de tamanho n. Para cada H_i, assim,

$$P(H_i|E) = P(H_i)\frac{P(E|H_i)}{P(E)} \tag{14}$$

onde: P(E) é a probabilidade a priori de uma evidência. $P(H_i)$ é a probabilidade a priori de uma hipótese. $P(H_i|E)$ é a probabilidade a posteriori de H_i, dado E. $P(E|H_i)$ é a probabilidade condicional inversa dá a posteriori.

O algoritmo seleciona a hipótese de maior probabilidade. Pelo fato de P(E) ser independente de qualquer hipótese, P(E) terá o mesmo valor para cada uma das hipóteses. Assim, pode-se simplificar a equação de modo a maximizar somente $P(E|H_i) \cdot P(H_i)$.

Uma rede bayesiana de crença é um modelo probabilístico que se apresenta como um grafo direcionado acíclico, no qual seus nós representam as evidências ou hipóteses, e seus arcos retratam suas relações de dependência entre os nós (COPPIN, 2010).

Cada nó possui uma tabela de probabilidades, que indicam as probabilidades de o evento ocorrer. Para nós que não tem pai (s), ou seja, nós que não possuam nenhuma aresta direcionada para o evento em questão, então a tabela mostra probabilidade *a priori*. Caso os nós contenham arestas direcionadas para o evento em questão, a tabela representa probabilidades condicionais.

O classificador Naive Bayes é baseado no teorema de Bayes, e consiste em um caso especial de redes bayesianas, onde a topologia é conhecida e as dimensões são fortemente independentes entre si, ficando somente dependentes da variável classe.

Apesar desse método de aprendizagem de máquina possuir fundamentações simples, apresenta uma alta taxa de classificação em problemas reais (HENKE *et al*., 2014).

Considerando várias dimensões (d_1,...,d_n) de uma instância a ser classificada, e um conjunto de C (c_1,...,c_n) classificações, apresenta-se

$$P(c_i|d_1, \dots, d_n) \tag{15}$$

onde c_i é a *i*-ésima classificação, no qual $c_i \in C$.

O algoritmo escolhe a maior probabilidade *a posteriori*, conhecida como máxima *a posteriori* ou hipótese MAP. Aplicando o teorema de Bayes nessa situação, obtêm-se

$$P(c_i|d_1, \dots, d_n) = \frac{P(d_1, \dots, d_n|c_i) \, . P(c_i)}{P(d_1, \dots, d_n)} \tag{16}$$

e devido a busca constantemente para encontrar a máxima *a posteriori*, e já que P(d_1, ... , d_n) é sempre independente de c_i, obtêm-se

$$P(d_1, \dots, d_n|c_i) \cdot P(c_i) \tag{17}$$

como o classificador naive bayes assume que todas as dimensões são independentes, a probabilidade inversa pode ser reescrita, na forma

$$P(c_i) \cdot \prod_{j=1}^{n} P(d_j|c_i) \tag{18}$$

As probabilidades *a priori* podem ser obtidas por meio da frequência em que as classes aparecem no conjunto de dados de treinamento. Essa característica de baixa complexidade na fase de treinamento torna naive bayes indicado para aplicações online (HENKE, et al., 2014).

Para esta técnica, foi realizado testes, mas não obteve os melhores resultados.

2.7.2 Árvore de Decisão

Árvore de decisão é um modelo baseado em árvore, que em termos formais, é um grafo acíclico direcionado, no qual utiliza a estratégia dividir para conquistar, a fim de resolver um problema de decisão (FACELI *et al*., 2011).

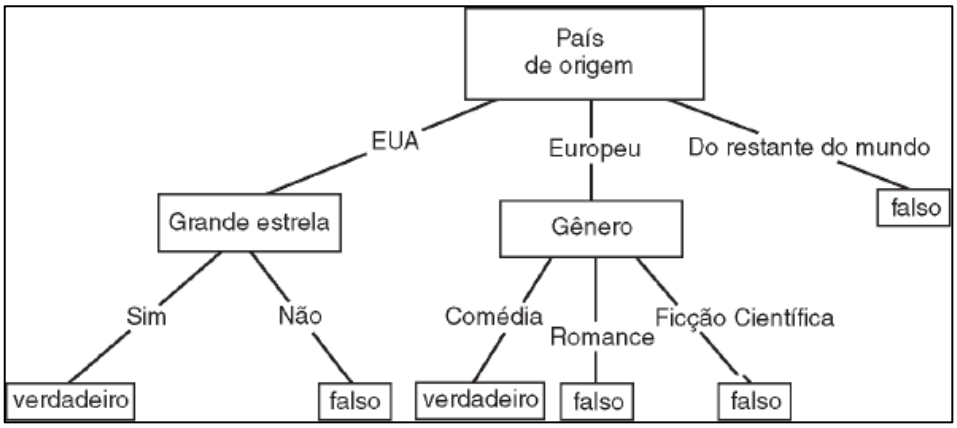
Conforme Quinlan (2014 apud ROLIM; MELLO; COSTA, 2017) árvore de decisão é uma técnica indutiva de aprendizado de máquina, que de acordo com os valores dos atributos de entrada, realiza uma predição.

Em uma árvore de decisão cada nó possui um teste lógico (com exceção dos nós folhas), do valor de uma característica do dado de entrada e suas arestas correspondem aos distintos valores possíveis das características. Os nós da árvore são nós de divisão, que possuem dois ou mais nós filhos e os nós folhas possuem sua respectiva classificação.

O procedimento de construção da árvore de decisão, se dá por meio do conjunto de dados de treinamento. O problema para construir uma árvore de decisão com o menor número de nós possíveis, condizente com um conjunto de dados, é de natureza NP completo (RIVEST, 1987 apud FACELI *et al*., 2011). Geralmente utiliza-se heurísticas para construção sem *back-tracking*, para alcançar um tempo linear em relação ao número de exemplos. No entanto, as soluções podem ser ótimas localmente, porém susceptíveis a soluções não ótimas globalmente.

Para se classificar uma nova instância em uma árvore de decisão, os atributos da amostra são submetidos ao um teste lógico no nó raiz, e dependendo do resultado, é selecionado um dos nós filhos, onde esse procedimento é repetido até alcançar algum nó folha para obter a classificação para a instância em questão. A Figura 6 ilustra uma árvore de decisão, onde determina se um dado filme será bem-sucedido ou não, em termos de bilheteria.

Figura 6 – Árvore de decisão simples para classificar se um dado filme terá sucesso de bilheteria.



Fonte: Coppin (2010).

Dado um exemplo de filme com suas características, o processo de classificação inicia no topo da árvore, no qual possui um teste lógico, se o filme foi feito nos Estados Unidos, escolhe-se o primeiro ramo, se foi feito na Europa, escolhe-se o segundo ramo, e se foi feito em outro lugar o filme não terá sucesso de bilheteria. Se o filme foi feito nos EUA e possui uma grande estrela terá sucesso de bilheteria.

Os algoritmos de árvore de decisão se diferem geralmente em suas métricas de decidirem qual teste de atributo é utilizado em cada nó. Os critérios de seleção são baseados em diferentes medidas, tais como a impureza, distância e dependência.

O ganho de informação está fundamentado na entropia. Entropia mede a aleatoriedade de uma variável alvo, ou seja, mede o quanto é difícil predizer determinado atributo alvo. A cada nó, é selecionado o atributo que mais reduz a entropia para dividir os dados. Para uma melhor compreensão, a definição informal de ganho de informação é a diferença da entropia do conjunto de dados de treinamento e a soma ponderada da entropia das partições (FACELI *et al.*, 2011).

Foi proposto a razão de ganho, com a finalidade de resolver os problemas enfrentados pelo ganho de informação. A razão de ganho é definida pela divisão entre o ganho de informação do atributo pela sua entropia, conforme a equação abaixo, na forma,

$$gainration(S, A) = \frac{gain(S, A)}{info(S, A)} \tag{19}$$

onde, S: quantidade total das amostras. A: atributo do conjunto de dados S. gain(S, A): ganho de informação do atributo A, contido no conjunto S. info(S, A): entropia do atributo A, contido no conjunto S.

O processo da razão de ganho consiste em realizar o ganho de informação de cada atributo, para efetuar o ganho médio. Na sequência, é selecionado apenas os atributos que obtiveram resultado maior do que o ganho médio, e por fim, é selecionado o atributo que melhor maximizar a razão de ganho.

Em domínios com ruídos, a poda de uma árvore é de extrema importância no momento da construção de uma árvore (FACELI *et al*., 2011). A poda consiste na substituição de nós profundos em nós folhas, visando uma melhor capacidade de generalização. Há dois problemas por causa dos ruídos (FACELI *et al*., 2011). O primeiro, é a classificação de um modo não confiável, pois folhas profundas caracterizam um número pequeno de exemplos, e estatisticamente mostrando que é pouco importante, refletindo um superajuste ao conjunto de dados de treinamento. O segundo problema, é que a árvore tende a ser profunda, dificultando o seu entendimento. A poda ajuda a minimizar esses problemas (FACELI *et al*., 2011).

Ao podar uma árvore, quase certamente irá ocorrer algumas classificações incorretas do conjunto de treinamento, no entanto, é evidente uma boa classificação para novas instâncias (FACELI *et al*., 2011). Há dois grupos de métodos para realizar a poda de uma árvore. A primeira é a pré-poda, onde a construção da árvore para ao satisfazer algum critério. E o pós-poda, efetua a poda depois que a árvore é construída. É evidente a desvantagem da pós-poda, ao desperdiçar a árvore construída, já que irá realizar a poda. Apesar da pós-poda ser mais lenta, é mais confiável (QUINLAN, 1988 apud FACELI *et al*., 2011).

Para esta técnica, foi realizado testes, porém não foi escolhido como modelo final, pois como já afirmado, por ser um método que se ajusta bem aos dados de treinamento, não consegue obter bons resultados em dados nunca vistos.

2.7.3 k-NN (k-*Nearest Neighbor*)

Métodos baseados em distância considera a proximidade entre os dados. A hipótese desses métodos, é que os dados similares tendem a estar próximos em uma região, consequentemente os não similares estarão distantes (FACELI *et al*., 2011).

O kNN é uma técnica de classificação baseado em instância, ou seja, consiste em armazenar o conjunto de treinamento e classificar instâncias novas, a partir de seus vizinhos mais próximos (HENKE *et al*., 2014).

A vizinhança é definida por uma medida de similaridade. Normalmente essa similaridade é a distância euclidiana, que é medida por

$$d(x, y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2} \tag{20}$$

onde, x é uma instância de treino, representado por $x = [\ x_1\ ,...,\ x_n\ ] \in R^n$ sendo n a dimensão do vetor de características, enquanto y é uma instância de teste, representado por $y = [\ y_1\ ,...,\ y_n\ ] \in R^n$.

No algoritmo kNN, o valor escolhido para o k, geralmente é ímpar, com o fim de evitar empates. Há duas estratégias para escolher o valor de k. A primeira é por validação cruzada, e a segunda é associar um peso à contribuição de cada vizinho.

O funcionamento do algoritmo é descrito abaixo, de acordo com Theodoridis (KOUTROUMBAS, 2006 apud HENKE *et al*., 2014).

1. Defina um valor para *k*, isto é, a quantidade de vizinhos que serão avaliados.
2. Calcule a distância da nova instância a ser classificada a todas as instâncias de treinamento.
3. Identifique os *k* vizinhos mais próximos da nova instância.
4. Conte o número de vizinhos mais próximos pertencente a cada classe.
5. Classifique a nova instância para a classe que possui a maior frequência nos *k* vizinhos mais próximos.

As vantagens do kNN são várias, como: o algoritmo de treinamento é simples, pois apenas armazena, pode ser aplicado a problemas complexos, é independente quanto a distribuição de dados no espaço de características.

No entanto, possui desvantagens quando os dados de treinamento são grandes, pois a complexidade computacional para classificação a fim de obter *k* vizinhos mais próximos torna um processo custoso. Assim como todo algoritmo baseado em distância, o kNN também sofre pela presença de atributos irrelevantes e redundantes (FACELI *et al*., 2011).

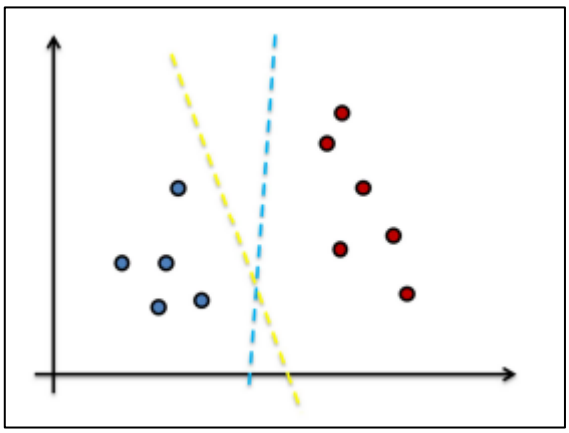
2.7.4 Máquina de Vetores de Suporte

As Máquinas de Vetores de Suporte (SVMs, do inglês, *Support Vector Machines*) são algoritmos supervisionados, que podem realizar tarefas de classificação ou regressão. São considerados eficientes por diminuir o *overfitting* e suportar muitas dimensões, por consequência menos suscetível à maldição da dimensionalidade (AMARAL, 2016).

O SVM possui uma boa capacidade de generalização dos dados, pois consegue separar bem os dados lineares como também os não lineares. Um conjunto de dados é dito linearmente separável, quando é possível encontrar padrões de classes distintas, em um hiperplano com ao menos duas dimensões e uma reta.

A Figura 7 mostra um conjunto de dados que possuem duas classes distintas: azuis e vermelhas. É possível separar as classes através de infinitas retas, porém o problema está em qual a melhor reta que produz uma melhor probabilidade de acerto em classificar classes em dados nunca vistos. Vale ressaltar que na Figura 7, quanto mais próxima a instância do centro mais complexo se torna a tarefa de classificação correta.

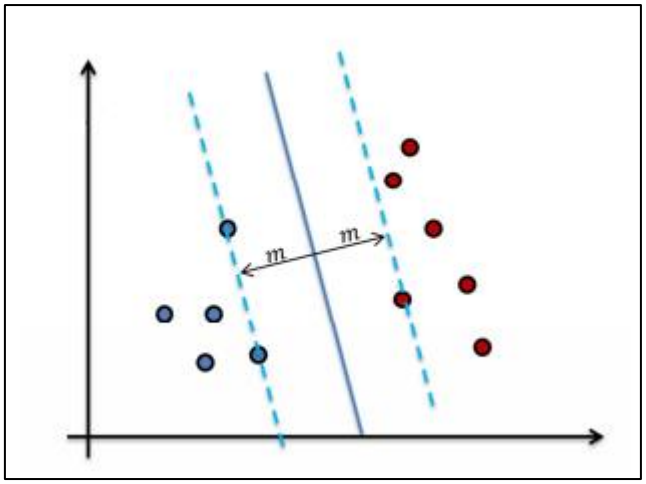
Figura 7 - Duas possíveis separações lineares de um conjunto de dados.



Fonte: Motta (2015).

O SVM busca encontrar o melhor hiperplano, no qual consiste em maximizar as margens de separação entre as classes, por meio de vetores de suporte, como ilustra a Figura 8. Os pontos sob as margens são chamados de vetores de suporte. E a reta ao centro é a referência para classificar novas instâncias.

Figura 8 – Hiperplano ótimo de separação entre duas classes de dados.

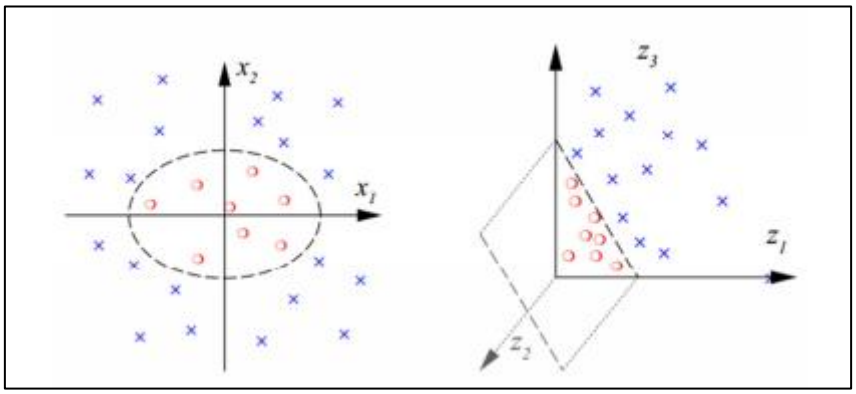


Fonte: Adaptado de Motta (2015).

Nem sempre os dados são linearmente separáveis, seja pela presença de *outliers*, ou pela própria natureza do problema, por não ser linear. Para ser possível traçar uma reta em distribuições de dados de classes não linearmente separáveis, é necessário suavizar as margens utilizando uma função de kernel, ou seja, deixar que alguns dados sejam classificados errados.

A Figura 9 ilustra uma distribuição de dados não lineares de entradas em duas dimensões, no qual são separáveis por um elipsoide. Transformando os dados utilizando os monômios de segunda ordem, os dados podem ser separáveis linearmente por um hiperplano.

Figura 9 – Exemplo de dados de duas dimensões. Utilizando a transformação, os dados da esquerda podem ser linearmente separáveis em um hiperplano.



Fonte: (Muller, *et al*. 2001 apud Motta, 2015)

A Tabela 2 mostra as funções de kernel mais utilizadas.

Tabela 2 – Funções kernel mais utilizadas.

RBF gaussiano	$\boldsymbol{k(x,y) = exp\left(\frac{-\|x-y\|^2}{c}\right)}$
Polinomial	$((x \cdot y) + \theta)^d$
Sigmoidal	$\tanh(k(x \cdot y) + \theta)$
Multiquádrico inverso	$\frac{1}{\sqrt{\|x-y\|^2 + c^2}}$

Fonte: (Muller, *et al*. 2001 apud Motta, 2015)

Para esta técnica, foi realizado testes, mas não obteve os melhores resultados.

2.8 APRENDIZADO NÃO SUPERVISIONADO

A forma de aprendizado não supervisionado aprende sem nenhuma intervenção, ou seja, não há nenhuma classificação pré-definida (COPPIN, 2010). O algoritmo tem que descobrir sozinho as relações entre os objetos, padrões, regularidades ou categorias nos dados. Um bom exemplo para mostrar o aprendizado não supervisionado, é o papel de um cientista. Ele não tem um auxílio como o de um professor, mas busca criar hipóteses para explicar fenômenos observados, avaliam e testam suas hipóteses por experimentos que eles mesmo concebem (LUGER, 2013).

Lenat construiu um dos primeiros programas a realizar descobertas, mesmo que não originais, chamado AM. Esse programa inicia com os conceitos de teoria dos conjuntos, operações para se criar um novo conhecimento a partir de modificações e heurísticas para detectar conceitos “interessantes”. O AM conseguiu a partir de conjuntos realizar somas, multiplicações como um processo de várias somas, e até mesmo a divisão de números inteiros. Com isso descobriu os números naturais e a existência de números primos, conceito importante da teoria dos números.

O AM não conseguia “aprender a aprender” e nunca desenvolvia uma compreensão profunda da matemática, e por isso Lenat construiu o EURISKO, que procurava aprender novas heurísticas (LUGER, 2013).

Vários outros pesquisadores desenvolveram programas para descoberta científica automática, como por exemplo Bacon, que ao desenvolver modelos computacionais a partir de distâncias dos planetas ao sol e o período das órbitas dos planetas, "redescobriu" as leis de Kepler do movimento planetário.

O avanço na descoberta científica autônoma tem sido modesto. Apesar da extrema importância para a ciência, o aprendizado não supervisionado tem enfatizado mais problemas de categorização. As categorias são essenciais para à cognição humana (LAKOFF, 1987 apud LUGER, 2013). Boa maioria do nosso conhecimento útil é sobre categorias, pois o modo de pensar não é em vacas específicas e sim em vacas em geral. Ao tentar explicar as causas das substâncias químicas reagirem da forma que fazem, a química se apoiou a grupos ou categorias como alcalinos e ácidos.

O aprendizado não supervisionado possui duas tarefas, que são as de agrupamento e regras de associação. O problema de agrupamento se dá por meio de um conjunto de instâncias não classificadas e uma forma de medir a semelhança entre elas. O objetivo principal é a formação de grupos dessas instâncias de modo a maximizar a semelhança entre os objetos.

As redes neurais artificiais também podem ser não supervisionadas. O mapa de Kohonen é uma rede neural capaz de agrupar instâncias sem receber quaisquer dados de treinamento pré-classificados. Donald Hebb, em 1949, propôs um método não supervisionado de redes neurais, conhecido como aprendizado de Hebb. O método consiste no princípio de que se uma entrada ativa dois neurônios (nós) conectados simultaneamente, essa conexão deve ser reforçada (COPPIN, 2010).

A tarefa de regras de associação busca encontrar elementos que se associem, ou seja, elementos que implicam na presença de outros elementos em uma mesma transação. São usadas métricas para avaliar a relevância das associações, que são suporte e confiança, como as principais. Imagine uma regra de implicação:

$$X \rightarrow Y \tag{21}$$

cujo X e Y são conjuntos de itens. Suporte é a proporção de transações que contém todos os itens, já confiança é a proporção de transações que, contendo X, também contém Y.

Existem vários algoritmos que produzem regras associativas como: *Apriori*, *FP-Grow*, dentre outros.

2.8.1 Clustering

Clustering, ou agrupamento em português, busca encontrar clusters (grupos) nos dados, de modo que, cada elemento pertencente ao cluster compartilhe de semelhanças entre elementos do mesmo cluster, por uma medida de similaridade pré-definida, que posteriormente esses grupos podem ser classificados (FACELI *et al*., 2011).

Em problemas que envolvem métodos de agrupamento, normalmente, não se encontra um atributo especial como a classe ou rótulo. Por essa razão, o aprendizado com algoritmos de agrupamento é categorizado como não supervisionado (AMARAL, 2016).

Existem vários problemas do mundo real, que pode ser aplicado tarefas de agrupamento, como: identificar grupos de clientes para direcionar campanhas, identificação de fraudes, agrupamento de documentos, classificar instâncias que não possuem classe conhecida, dentre outras.

Existem diferentes níveis de proximidade: entre objetos, entre objeto e o cluster e entre clusters (HE, 1999 apud FACELI *et al*., 2011).

A medida de proximidade entre objetos pode ser de similaridade ou dissimilaridade. Uma das medidas de similaridades mais usadas é a de correlação, enquanto a de dissimilaridade é a distância euclidiana.

As medidas mais utilizadas para atributos contínuos e racionais, são medidas de distância baseada nas métricas de Minkowski. Existem variações da métrica de Minkowski como: distância de Manhattan ou Hamming, distância euclidiana e a distância Chebyschev ou *supremum*.

Para atributos nominais, a medida mais utilizada é a distância de Hamming. E para conjunto de dados que apresentam dados de diferentes tipos ou heterogêneos, pode-se utilizar uma composição das medidas euclidiana e Hamming ou o coeficiente geral de similaridade.

Os tipos de algoritmos de agrupamento podem ser particionais, hierárquico, *fuzzy* (difuso), completos, baseado em protótipos, baseado em densidade (AMARAL, 2016).

Os métodos particionais, dividem o conjunto de dados em *k*-grupos, de acordo com a medida de proximidade, onde *k* é dado pelo usuário. O algoritmo seleciona *k* objetos para serem o centro dos grupos, e então utiliza uma estratégia iterativa para mudar os objetos de grupos. Após a divisão inicial, há duas possibilidades para selecionar o objeto que irá ser a referência para a medida de proximidade. A primeira é utilizando a média do *cluster* em questão, conhecido também como centro de gravidade do cluster, o algoritmo *k-means* utiliza essa abordagem. A segunda é selecionar o objeto mais próximo ao centro de gravidade, o algoritmo *k-medoid* utiliza essa abordagem.

Os métodos hierárquicos agrupam o conjunto de dados em uma estrutura hierárquica. Os resultados do agrupamento são representados, normalmente, em forma de dendrograma. Há duas formas de abordagem: aglomerativa e divisiva.

Na abordagem aglomerativa, cada instância é um *cluster*, onde calcula-se a distância entre cada um deles, armazena em uma matriz, e junta os dois cluster que estão mais próximos. Esse processo de repete até formar apenas um *cluster*.

O inverso ocorre na abordagem divisiva, onde começa todas as instâncias em um único cluster, e o processo realiza a divisão até que se obtenha n clusters ou o *k* desejado aconteça.

2.9 APRENDIZADO POR REFORÇO

O ser humano por muita das vezes, aprende com as interações com o mundo. E não é diferente do aprendizado por reforço, que consiste em receber um reforço positivo ao agir corretamente e consequentemente um reforço negativo ao agir incorretamente (COPPIN, 2010). Não se diz o porquê ou como agiu corretamente, apenas informa que a tarefa foi realizada. O agente recebe o reforço e armazena essas informações e utiliza nas escolhas das próximas ações.

O aprendizado por reforço não tem uma abordagem supervisionada, mas busca através de tentativa, erro e realimentação construir suas próprias hipóteses para atingir um objetivo no ambiente no qual está situado (LUGER, 2013).

Uma característica importante para o aprendizado por reforço, é explorar partes desconhecidas do seu ambiente, pois ao usar apenas o que sabe, não conseguirá resultados mais eficientes, consequentemente não terá sucesso.

Existe quatro componentes do aprendizado por reforço, que são: uma política, uma função de recompensa, uma função de valor e um modelo do ambiente (LUGER, 2013).

A política é um componente crítico, já que é suficiente pelo comportamento do agente em um dado tempo. A política pode ser representada por um conjunto de regras de produção ou uma simples tabela de consulta, já que ela é responsável pelas ações e método de agir do agente. A política do agente deve escolher ações que maximizem o valor final dos reforços recebidos durante um intervalo de tempo.

A função de valor é uma propriedade de cada estado do ambiente, que indica para o agente o que ele pode esperar de recompensa ao longo prazo se feitas ações a partir daquele estado. A função de recompensa mede o quanto é desejável no dado momento agir de determinada maneira.

O modelo é opcional em um agente que aprende por reforço. O modelo mapeia aspectos do comportamento do ambiente. Serve para determinar falhas assim como determinar um plano de ações.

Há três famílias de algoritmos de inferência por aprendizado por reforço: aprendizado por diferença temporal, programação dinâmica e métodos Monte Carlo (SUTTON; BARTO, 1998 apud LUGER, 2013).

No aprendizado por diferença temporal, os métodos são iterativos, durante as interações do agente com o ambiente. A cada interação com o ambiente, o algoritmo reduz gradativamente a diferença entre expectativa da recompensa e a recompensa recebida do ambiente, com base na atualização das funções valor.

Os métodos de programação dinâmica calculam funções de valor retornando valores de estados sucessores para estados predecessores.

Os métodos de Monte Carlo consideram todos os pares estado-ação, atualizando ao fim de cada iteração com o ambiente. Para problemas onde a propriedade de Markov não é satisfeita, os métodos de Monte Carlo são mais apropriados para tarefas de aprendizado.

2.10 MÉTODOS ENSEMBLE

Os modelos *ensemble*, são formados por um conjunto de preditores, no qual suas decisões são combinadas, a fim de obter uma melhor decisão na tarefa submetida (DIETTERICH, 1997 apud FACELI *et al*., 2011).

A ideia para afirmar que modelos que trabalham em conjunto geralmente são melhores do que um individual, é muito simples. A probabilidade de várias pessoas, ou especialistas de domínio, ou preditores errarem uma decisão é menor do que a probabilidade de apenas uma pessoa, especialista ou preditor errar.

No entanto, se os preditores forem idênticos, a técnica se torna inútil. É preciso que os preditores tenham um nível substancial de desacordo. Vale ressaltar, que os classificadores têm de fazer previsões de forma independente, ou seja, realizar predições não correlacionadas (FACELI *et al*., 2011).

Com relação a combinação das predições, pode-se distinguir em duas formas: métodos de votação e métodos de seriação (FACELI *et al*., 2011). Onde no primeiro método, a saída da predição é uma classe, e do método de seriação é uma probabilidade para cada possível classe.

Pode-se diferenciar entre métodos dinâmicos e métodos estáticos. Se leva em consideração o exemplo de teste, no qual os métodos estáticos combinam todos os resultados de predições, e os métodos dinâmicos, seleciona os melhores classificadores para aquele determinado exemplo de teste.

Os métodos por votação podem ser uniforme, onde cada classificador possui a mesma importância para o resultado final, e o outro tipo é o por peso, no qual classificadores possuem pesos diferentes, influenciando assim o resultado final predito. Os métodos por seriação podem melhorar a votação uniforme, em que seria apresentado a estimativa da probabilidade de cada classe.

2.10.1 Combinação de Classificadores Homogêneos

A combinação de classificadores homogêneos realiza a combinação de modelos em um único algoritmo. Há diversas estratégias para geração de modelos com distintas hipóteses, haja vista, que a diversidade entre os modelos é de extrema importância para a eficácia dos métodos

ensemble. A maioria dos métodos utiliza o conjunto de treinamento para diversificar as hipóteses, utilizando amostras distintas do conjunto de treinamento.

Para gerar diversidade em classificadores, existem vários métodos, como: por amostragem dos objetos e atributos, injeção de aleatoriedade, e perturbação dos exemplos de teste (FACELI *et al*., 2011).

Uma das técnicas baseada em amostragem de exemplos de treinamento é o *Bootstrap Aggregating – bagging*, este método consiste em dividir os dados em amostras *bootstrap*, e criar preditores com essas amostras, e combiná-los. Caso o problema fosse de classificação, utilizaria um sistema de votação e se o problema fosse de característica de regressão, geralmente usa-se a média. Um algoritmo que utiliza essa técnica é o *Random Forest*.

Outra técnica surgiu na comunidade de *machine learning*, no qual a pergunta em termos informais era: poderá um conjunto de classificadores fracos (capacidade de generalização é um pouco melhor do que a escolha aleatória) gerar um classificador forte?

A resposta para essa pergunta, foi a técnica de *Boosting*, que consiste em atribuir um peso para cada exemplo do conjunto de dados, gerando classificadores distintos. Essa técnica é iterativa, onde os pesos são ajustados com base nos classificadores anteriores até a dada iteração. Há diversos algoritmos que utiliza essa técnica, como: *AdaBoost*, *LogitBoost* e *GradientBoosting*.

Neste trabalho, esta técnica de métodos em conjunto foi utilizada, tendo em vista as que obtiveram os melhores resultados.

3 ANÁLISE PREDITIVA DE INTERAÇÕES PROTEÍNA-PROTEÍNA

Neste capítulo, na seção 3.1 é apresentado a análise dos resultados, onde foi realizado diversos testes com a utilização de validação cruzada, efetuado ajustes nos hiperparâmetros do algoritmo, e avaliado o modelo, a fim de obter a satisfação do modelo criado.

3.1 ANÁLISE DOS RESULTADOS

Após realizar o pré-processamento no conjunto de dados SalivaTecDB, ficaram 1.826 proteínas de bactérias presentes na saliva, 3.019 proteínas humanas encontradas na saliva, na qual totalizando suas interações no universo do BioGrid, correspondem a 36.994 interações positivas. O Apêndice A mostra parcialmente o código-fonte do pré-processamento realizado.

O conjunto de dados no qual possíveis proteínas não interagem entre si, foi formado por 36.049 interações negativas. O total de interações positivas e negativas foram 73.043 exemplos.

Devido à grande complexidade dos problemas de mineração de dados, a melhor prática para encontrar um possível modelo para a solução, é realizando vários experimentos e testes para diferentes algoritmos, aplicando diversos parâmetros (DEAN, 2014 apud SERRAS, 2015).

Ao identificar a tarefa de aprendizado como classificação binária supervisionada (o par de proteína interage ou não interage), efetuar a construção e integração dos banco de dados (BioGrid, SalivaTecDB e UniProt), realizar o pré-processamento necessário e transformação dos dados, foi aplicado a validação cruzada de 5-*folds* estratificados, na qual consiste em dividir todo o conjunto de dados em cinco partes aproximadamente iguais e que tenham a mesma proporção de classes do conjunto de dados original, cada uma das partições são usadas como teste, enquanto as outras quatro partições são usadas para treinamento, até o fim do ciclo de cinco iterações. Onde o resultado é a média de acurácia das cinco iterações.

A escolha para cinco partições na validação cruzada se deu a estudos onde são apresentadas evidências empíricas de que a escolha para o *k* ser cinco ou dez, produz um *trade-off*[19] razoável entre viés e variância (KOHAVI, 1995 apud MOSS; LESLIE; RAYSON, 2018).

O processo de validação cruzada foi realizado em seis algoritmos de classificação supervisionada, onde os dois melhores ranqueados pertencem a categoria formada por um

[19] Termo da língua inglesa para representar uma situação em que há conflito de escolha, um "perde-e-ganha", ou seja, a resolução de um problema acarreta em outro.

comitê de classificadores. A Tabela 3 mostra o desempenho de cada um dos algoritmos em relação a acurácia, assim como o respectivo desvio padrão.

Tabela 3 – Desempenho dos algoritmos com validação cruzada 5-folds.

ALGORITMO	MÉDIA DE ACURÁCIA	DESVIO PADRÃO
RANDOM FOREST	0.792013	0.005742
GRADIENT BOOSTING	0.770231	0.002319
ÁRVORE DE DECISÃO	0.720822	0.001814
ADA BOOST	0.695084	0.005627
SVM	0.694673	0.004266
GAUSSIAN NAIVE BAYES	0.597196	0.003367

Fonte: Própria autoria (2018).

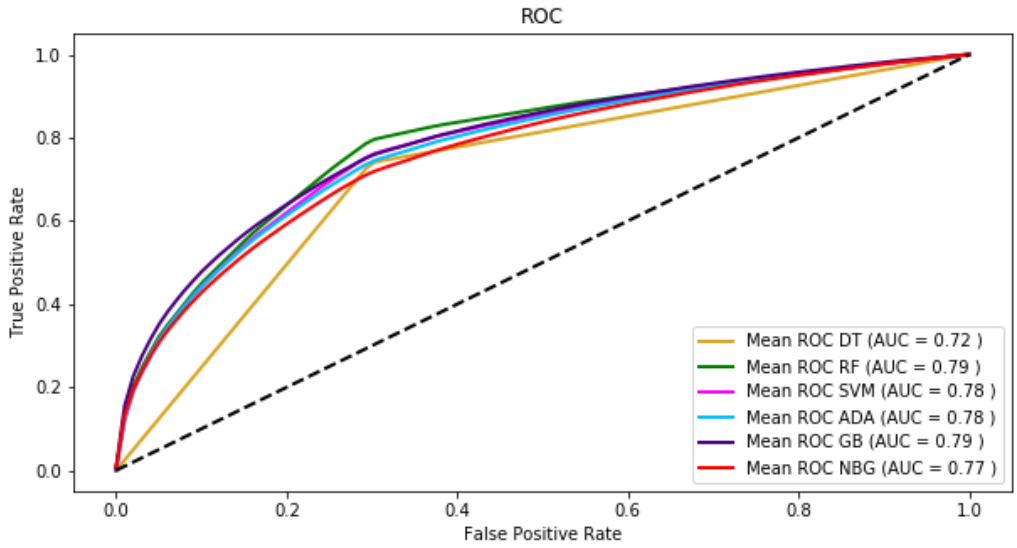
O fato de pertencer a essa categoria, confirma ainda mais o que a literatura vem afirmando sobre os algoritmos em conjunto, por obterem um melhor desempenho em relação a classificadores solitários.

Quanto ao desvio padrão, a Árvore de Decisão e o *Gradient Boosting* obtiveram os menores resultados, porém todos os algoritmos obtiveram desvios padrões relativamente baixos, o que indica pouca variabilidade em relação à média dos resultados dos 5-*folds*, e uma estabilidade perante mudanças nos exemplos, ou seja, a sensibilidade é pequena a poucas alterações nos exemplos do conjunto de treinamento.

Geralmente, métricas escalares como a acurácia, nem sempre representam a realidade. Principalmente quando as classes são desbalanceadas. E por muitas vezes é preciso validar modelos de forma a visualizar toda a sua trajetória com relação a sensibilidade (em outras literaturas é conhecida como *recall*) e especificidade. A curva ROC permite realizar isso, fornecendo uma estimativa do desempenho do classificador para diferentes variações.

A área sob a curva ROC é uma boa métrica escalar para representar a realidade como um todo. O Gráfico 1 mostra a média das curvas ROC dos 5-*folds* de cada validação cruzada aplicada aos algoritmos.

Gráfico 1 – Média das curvas ROC de cada modelo.



Fonte: Própria autoria (2018).

O eixo das abscissas representa a taxa de falsos positivos, e o eixo das ordenas a taxa de verdadeiros positivos. Todos os classificadores ficaram acima da linha diagonal, o que significa que todos possuem desempenhos melhores do que uma função aleatória. Os dois algoritmos que obtiveram uma maior área sob a curva ROC foram o *Random Forest* representado pela cor verde e *Gradient Boosting* representado pela cor roxo.

Após ter conhecimento dos dois algoritmos que obtiveram melhor desempenho em acurácia e área sob curva ROC, o conjunto de dados foi dividido em dois momentos, o primeiro momento com 70% para dados de treinamento e ajustes dos hiperparâmetros e 30% para teste, e segundo momento com 80% para treinamento e 20% para teste. Utilizou-se nos dois momentos os dois melhores algoritmos ranqueados.

Vale ressaltar que, os dois momentos escolhidos de 70% e 80% para dados de treinamento e 30% e 20% para dados de teste, a motivação para tal escolha, é por serem os valores mais comuns na literatura, outro fator, é que de acordo com a quantidade de dados de treinamento e dados de teste, e variabilidade dos dados, os resultados podem variar bastante.

Após os ajustes nos hiperparâmetros, e validação com 30% dos dados, o classificador *Gradient Boosting* obteve uma acurácia de pouco mais de 84% (0,8433), e classificador *Random Forest* foi obtido uma acurácia de pouco mais de 78% (0,7839).

No segundo momento, com 20% dos dados para teste, o *Gradient Boosting* obteve 84% (0,8450) de acurácia. Já o *Random Forest* teve 78% (0,7834) de acurácia.

Com os ajustes dos hiperparâmetros, o *Gradient Boosting* obteve nos dois momentos obteve um aumento de 7% em acurácia. Já o *Random Forest* houve uma queda de 1%. Assim, fica evidente que o *Gradient Boosting* obteve um melhor desempenho em relação ao *Random Forest*. É possível observar que a variabilidade dos resultados dos dois momentos, tanto o *Gradient Boosting* como o *Random Forest*, obtiveram diferenças muito pequenas, o que evidencia uma certa estabilidade.

No processo de afinação dos hiperparâmetros, foi escolhido um método exaustivo (mais conhecido como *Grid Search*) combinado em uma validação cruzada de cinco partes, ou seja, o método calcula uma estimativa de desempenho para a validação cruzada em um conjunto de possíveis valores de parâmetros (grade), então é selecionada a melhor combinação que melhor maximize a acurácia. O Apêndice B mostra o conjunto usado para a grade. A Tabela 4 mostra a calibração ótima para o classificador *Gradient Boosting*.

Tabela 4 – Parâmetros ótimos do classificador *Gradient Boosting*.

PARÂMETRO	VALOR ÓTIMO
TAXA DE APRENDIZADO	0.2
MÁXIMO DE CARACTERÍSTICAS	sqrt
PROFUNDIDADE MÁXIMA	5
NÚMERO DE ESTIMADORES	300

Fonte: Própria autoria (2018).

Vale ressaltar, que taxa de aprendizado, significa a contribuição de cada árvore, máximo de características (*features*) é o número máximo de características considerados na construção de uma dada árvore, profundidade máxima indica a profundidade máxima de cada árvore e número de estimadores significa o número de árvores construídas pelo algoritmo.

Geralmente uma taxa de aprendizado baixa obtém melhores resultados, porém o fato de a escolha ser relativamente alta para a taxa de aprendizado, se dá a existência de um *trade-off* entre a taxa de aprendizado e o número de estimadores por ser relativamente alto. A raiz quadrada do número de dimensões como *max features* não obteve surpresa, pois é considerada pela literatura como a que obtém melhores resultados. A Tabela 5 mostra a matriz de confusão do Gradient Boosting com 30% para dados de teste.

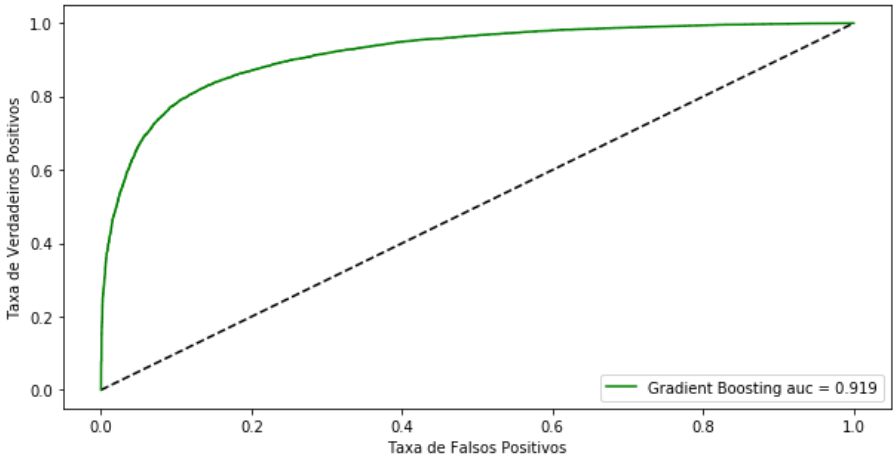
Tabela 5 – Matriz de confusão do *Gradient Boosting* com ajustes afinados, com 30% de dados de teste.

	CLASSE PREDITA	
CLASSE VERDADEIRA	*POSITIVO*	*NEGATIVO*
POSITIVO	9322	1591
NEGATIVO	1842	9158

Fonte: Própria autoria (2018).

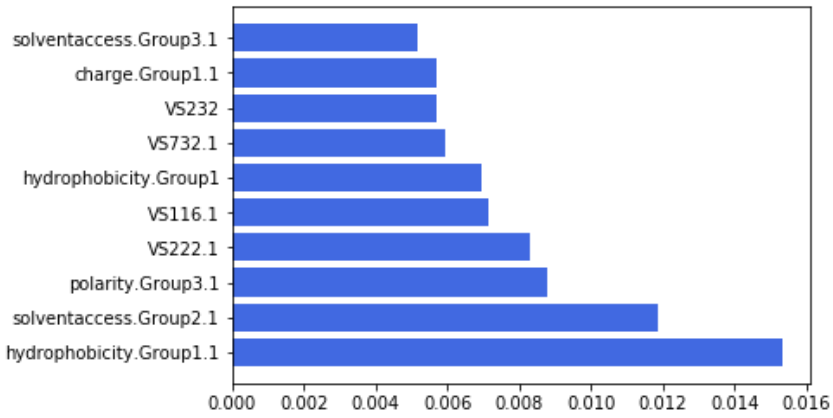
O Gráfico 2 apresenta a curva ROC do *Gradient Boosting*. É facilmente identificável a melhoria de desempenho em relação a todos os seis algoritmos testados inicialmente. A curva chega próximo a coordenada (0,1). Isso significa que o modelo obteve uma boa generalização dos dados. Vale ressaltar que, a área sob curva ROC obteve um ótimo resultado, com um escalar de 0.919.

Gráfico 2 – Curva ROC do *Gradient Boosting*.



O algoritmo Gradient Boosting elegeu a característica hidrofobicidade como sua característica mais importante. O Gráfico 3 mostra um gráfico de barras que possui as dez *features* mais importantes.

Gráfico 3 – As dez features mais importantes para o *Gradient Boosting* com 70% dos dados.



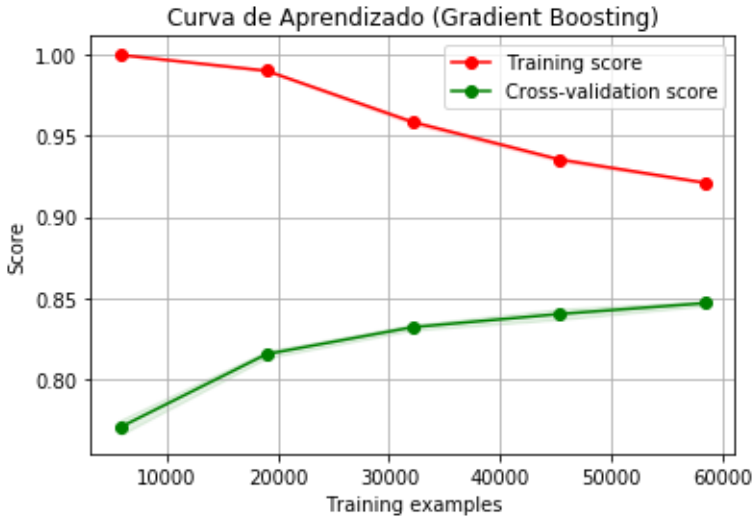
Fonte: Própria autoria (2018).

O fato de a escolha ser hidrofobicidade de categoria polar, não se caracteriza uma surpresa, pois proteínas que possuem suas cadeias laterais átomos de nitrogênio e oxigênio são capazes de estabelecerem ligações de hidrogênio com as moléculas de água.

A curva de aprendizado permite avaliar o viés e variância do modelo, identificando assim, se o modelo sofre de alto viés ou alta variância, se for o caso. O Gráfico 4 mostra a curva de aprendizado do modelo *Gradient Boosting*.

É fato que na curva de aprendizado do *Gradient Boosting*, quanto menos exemplos no treinamento o seu erro tende a diminuir, porém nos dados de validação o erro aumenta. Conforme o número de exemplos aumenta, os erros de treinamento começam a aumentar, porém o modelo começa a acertar em dados nunca vistos. A curva expressa que se houver mais exemplos, é provável que o modelo aumente ainda mais sua acurácia.

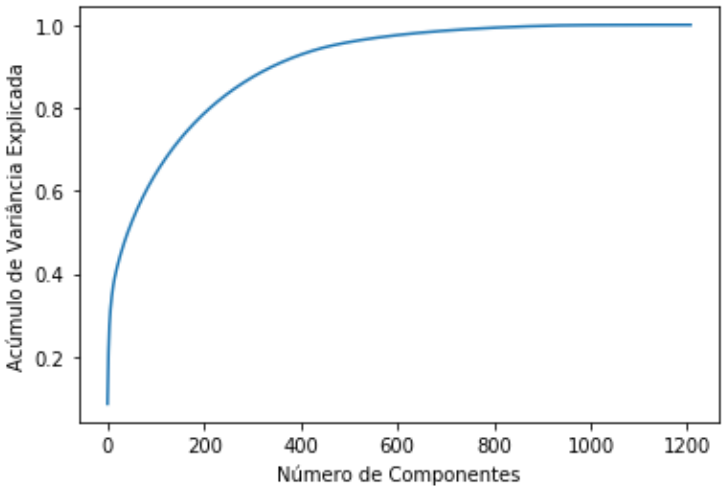
Gráfico 4 – Curva de Aprendizado do *Gradient Boosting* com hiperparâmetros ajustados.



Fonte: Própria autoria (2018).

Após essa bateria de experimentos e testes, foi utilizado a redução de dimensionalidade, a fim de obter uma melhor acurácia se for o caso, ou uma acurácia próxima, porém com menos complexidade, ou seja, menos características. Uma parte vital da Análise de Componentes Principais (PCA) é a capacidade de utilização para estimar quantos componentes principais é necessário para descrever os dados. Isso pode ser observado com a utilização de uma função entre a taxa de acumulação de variação explicada e o número de componentes principais. O Gráfico 5 mostra essa função.

Gráfico 5 – Nível de explicação dos dados de acordo com o número de componentes principais.



Fonte: Própria autoria (2018).

É fácil observar que a partir de 800 componentes principais a curva tende a ficar constante, no qual indica, que a utilização de mais componentes principais obtém-se um aumento pouco significativo para a explicação do conjunto de dados.

Com a utilização de 800 componentes principais, o *Gradient Boosting* com os hiperparâmetros ajustados, obteve uma acurácia de pouco mais de 77% (0,7792). O que mostra que foi bem abaixo do esperado. A Tabela 6 mostra a acurácia do *Gradient Boosting* com distintos números de componentes principais.

Tabela 6 – Desempenho do *Gradient Boosting* com diferentes números de componentes principais.

Nº DE COMPONENTES	ACURÁCIA
40	0.7168
300	0.7643
500	0.7679
800	0.7792

Fonte: Própria autoria (2018).

Percebe-se que no intervalo de 300 e 800 números de componentes principais, a acurácia do modelo obteve pouco aumento.

4 CONSIDERAÇÕES FINAIS

Esta pesquisa objetivou realizar uma análise computacional preditiva, entre pares de interação entre proteínas presentes na saliva oral humana, no universo do BioGrid e SalivaTecDB, onde foi construído um modelo preditivo com ótimos resultados.

Neste trabalho, foi obtido todos os dados necessários para a integração e construção do conjunto de dados (mapa de interações entre proteínas presentes na saliva da cavidade oral humana, com base no BioGrid), assim como seu pré-processamento e transformação.

Foi construído o modelo preditivo e validado com métricas baseadas na matriz de confusão, como por exemplo a acurácia, área sob curva ROC e curva de aprendizado.

As interações proteína-proteína são essenciais para os processos biológicos. O estudo das redes de interações de proteínas tem sido impulsionado por aplicações como a descoberta de novos medicamentos. No entanto, métodos experimentais capazes de verificar PPIs são caros, demorados e cobrem apenas uma fração das redes completas de PPIs. Portanto, este trabalho teve como principal contribuição a criação de um modelo preditivo para classificar interações entre proteínas presentes na saliva oral humana.

Este trabalho trouxe evidências de que os métodos de classificadores em conjunto possuem melhores desempenhos em relação a classificadores solitários. Realçou a necessidade de ajustes nos parâmetros do algoritmo. Também confirmou a importância de aminoácidos com características de hidrofobicidade de categoria polar na interação entre proteínas.

O pouco entendimento sobre bioquímica dificultou um andamento inicial da pesquisa mais rápida. Um dos mais importantes fatores de dificuldade é a parte de extração de atributos de sequências para reconhecimento de padrões de interação, onde atualmente existem poucas bibliotecas que facilitam a extração. Outro fator são os poucos trabalhos relacionados na língua nativa. As dificuldades até aqui citadas, foram solucionadas.

O fato de não processar os algoritmos em nuvem, com a possibilidade de utilização de vários computadores em processamento paralelo, dificultou a utilização de vários outros algoritmos, melhores ajustes nos hiperparâmetros e a utilização de um kernel PCA para aumentar a dimensão caso a distribuição dos dados não seja linearmente separável. Esta dificuldade não foi solucionada, por falta de tempo hábil.

Como sugestões para possíveis trabalhos futuros, buscar adicionar mais *features* ao conjunto de dados, por meio de métodos baseados em dados genômicos (vizinhança de genes,

perfis filogenéticos), estrutura proteica (homologia, encadeamento), informação de domínio, ontologia gênica, dentre outros.

Utilizar redes neurais profundas para verificar seu desempenho em relação aos métodos de *machine learning*. Processar um conjunto maior de possíveis ajustes de parâmetros, utilizar kernel PCA, além de processar os algoritmos em nuvem com processamento paralelo.

REFERÊNCIAS

AMARAL, F. **Introdução à ciência de dados**: mineração de dados e Big Data. 1. ed. Rio de Janeiro: Alta Books, 2016.

CERVO, A. L.; BERVIAN, P. A.; SILVA. R. **Metodologia científica**. 6. ed. São Paulo: Pearson Prentice Hall, 2007.

CHUANG, H. Y. et al. Network-based classification of breast cancer metastasis. **Molecular Systems Biology**, Oct. 2007. Disponível em: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2063581/>. Acesso em: 19 mar. 2018.

COELHO, E. D. et al. Computational prediction of the human-microbial oral interactome. **BMC Systems Biology**, Aug. 2014. Disponível em: <http://www.biomedcentral.com/1752-0509/8/24>. Acesso em: 19 mar. 2018.

COPPIN, B. **Inteligência Artificial**. 1. ed. Rio de Janeiro: LTC, 2010.

FACELI, K. et al. **Inteligência Artificial**: Uma Abordagem de Aprendizagem de Máquina. 1. ed. Rio de Janeiro: LTC, 2011.

GOLDBARG, M. C.; GOLDBARG, E. **Grafos**: conceitos, algoritmos e aplicações. 1. ed. Rio de Janeiro: Elsevier, 2012.

HENKE, M. et al. Aprendizagem de Máquina para Segurança em Redes de Computadores: Métodos e Aplicações. **In book: Livro de Minicursos do XI Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais**. Set, 2014. Disponível em: <https://www.researchgate.net/publication/228447003_Aprendizagem_de_Maquina_para_Seguranca_em_Redes_de_Computadores_Metodos_e_Aplicacoes>. Acesso em: 23 abr. 2018.

ITO, T. et al. A comprehensive two-hybrid analysis to explore the yeast protein interactome. **National Academy of Sciences**, Mar. 2001. Disponível em: < http://www.pnas.org/content/98/8/4569>. Acesso em: 19 mar. 2018.

LACOUNT, D. J. et al. A protein interaction network of the malaria parasite Plasmodium falciparum. **Nature**, Nov. 2005. Disponível em: <https://www.nature.com/articles/nature04104>. Acesso em: 19 mar. 2018.

LIMA, R. A. F. **Estratégias de seleção de atributos para detecção de anomalias em transações eletrônicas**. Belo Horizonte, 2016. Disponível em: <https://www.dcc.ufmg.br/pos/cursos/defesas/1930M.PDF>. Acesso em: 10 abr. 2018.

LUGER, G. F. **Inteligência Artificial**. 6. ed. São Paulo: Person Education do Brasil, 2013.

MOSS, H. B.; LESLIE, D. S.; RAYSON, P. Using J-K-fold Cross Validation to Reduce Variance When Tuning NLP Models. **Coling**, Jun. 2018. Disponível em: < https://arxiv.org/abs/1806.07139>. Acesso em: 09 nov. 2018.

MOTTA, S. A. C. S. **Detecção de falhas em dados sísmicos 3D utilizando funções geoestatísticas e SVM**. São Luís, 2015. Disponível em: < http://tedebc.ufma.br:8080/jspui/handle/tede/286>. Acesso em: 07 nov. 2018.

NASCIMENTO, A. C. A. **Combinação de kernels para predição de interações em redes biológicas**. Recife, 2015. Disponível em: <https://repositorio.ufpe.br/handle/123456789/16781>. Acesso em: 19 mar. 2018.

PRODANOV, C. C.; FREITAS, E. C. **Metodologia do trabalho científico**: métodos e técnicas da pesquisa e do trabalho acadêmico. 2. ed. Novo Hamburgo: Freevale, 2013.

RAMOS, J. A. P. **Árvores de Decisão Aplicadas À Detecção de Fraudes Bancárias**. Brasília, 2014. Disponível em: < http://repositorio.unb.br/bitstream/10482/16954/1/2014_JoseAbiliodePaivaRamos.pdf>. Acesso em: 23 abr. 2018.

RIGAUT, G. et al. A generic protein purification method for protein complex characterization and proteome exploration. **Nature Biotechnology**, Oct. 1999. Disponível em: <https://www.nature.com/articles/nbt1099_1030>. Acesso em: 19 mar. 2018.

ROLIM, V. B.; MELLO, R. F. L.; COSTA, E. B. Utilização de Técnicas de Aprendizado de Máquina para Acompanhamento de Fóruns Educacionais. **Revista Brasileira de Informática na Educação – RBIE Brazilian Journal of Computers in Education**. Oct, 2017. Disponível em: <www.br-ie.org/pub/index.php/rbie/article/download/7118/5537>. Acesso em: 23 abr. 2018.

RUSSELL, S. J.; NORVIG, P. **Inteligência Artificial**. 3. ed. Rio de Janeiro: Elsevier, 2013.

SANTOS, K. N. **Utilização de Técnicas de Aprendizado de Máquina para Predição de Crises Epiléticas**. Natal, 2016. Disponível em: < https://repositorio.ufrn.br/jspui/bitstream/123456789/24209/1/KelysonNunesDosSantos_DISSERT.pdf>. Acesso em: 23 abr. 2018.

SERRAS, F. A. J. **Métodos de Aprendizagem Automática:** um estudo baseado na avaliação e previsão de clientes bancários. Lisboa, 2015. Disponível em: <https://run.unl.pt/bitstream/10362/17371/1/TGI0051.pdf>. Acesso em: 07 nov. 2018.

SOARES, A. N. G. **Interatómica da cavidade oral**. Viseu, 2014. Disponível em: < http://hdl.handle.net/10400.14/16160>. Acesso em: 19 mar. 2018.

TONG. A. H. Y. et al. Systematic genetic analysis with ordered arrays of yeast deletion mutants. **American Association for the Advancement of Science**, Dec. 2001. Disponível em: < http://science.sciencemag.org/content/294/5550/2364.full>. Acesso em: 19 mar. 2018.

TURING, A. M. Computing Machine and Intelligence. **Mind**, Oct. 1950. Disponível em: <https://academic.oup.com/mind/article/LIX/236/433/986238>. Acesso em: 05 abr. 2018.

VERLI, H. et al. **Bioinformática**: da Biologia à Flexibilidade Molecular. 1. ed. São Paulo: SBBq, 2014.

VIEIRA, R. S. **Estudo da predição de Hot Spots em complexos Proteína-Proteína**. Santo André, 2013.Disponível em: <http://bdtd.ibict.br/vufind/Record/UFBC_c041f61ed931148e6879f38c72158912>. Acesso em: 12 nov. 2018.

YANG, Y.; WANG, H.; ERIE, D. A. Quantitative characterization of biomolecular assemblies and interactions using atomic force microscopy. **Methods**, Feb. 2003. Disponível em: < http://europepmc.org/abstract/MED/12606223>. Acesso em: 19 mar. 2018.

APÊNDICE A – CÓDIGO-FONTE PARCIAL DO PRÉ-PROCESSAMENTO

```
#Importação das bibliotecas necessárias
import pandas as pd
import numpy as np
from Bio.Seq import Seq
from Bio import SeqIO

#Importação do conjunto de dados
protBactOrais = pd.read_excel('proteinas_orais.xlsx')

#Deletando instâncias que contenham o organismo de categoria humana
protBactOrais.drop(protBactOrais[(protBactOrais['Organism'] == 'Homo sapiens (Human)') |
(protBactOrais['Organism'] == 'Homo sapiens')].index, inplace=True)

#Removendo instâncias duplicadas
protBactOrais.drop_duplicates(['UniProtKBAC'], inplace=True)

#Montando um dicionário com chave: código UniProt e valor: sequência primária da proteína
dicFasta = {}
for z in SeqIO.parse("uniprot_sprot.fasta", "fasta"):
    codUniProt = z.description.split('|')
    dicFasta[codUniProt[1]] = str(z.seq)
```

APÊNDICE B – GRADE DE VALORES PARA BUSCA EXAUSTIVA DO GRADIENT BOOSTING

PARÂMETROS	VALORES
TAXA DE APRENDIZADO	0.0001 - 0.001 - 0.01 - 0.05 - 0.1 - 0.15 - 0.2
PROFUNDIDADE MÁXIMA	3 - 5
MÁXIMO DE CARACTERÍSTICAS	log2 - sqrt
NÚMERO DE ESTIMADORES	10 - 100 - 250 - 300