Instituto Federal de Educação, Ciência e Tecnologia de Brasília, campus Taguatinga,
Campus Taguatinga

EXTRAÇÃO DE CARACTERÍSTICAS EM SNORNAS USANDO MODELOS MATEMÁTICOS

Por

MARCOS BEZERRA CAMPOS

Trabalho de Graduação

BRASÍLIA/2023

Marcos Bezerra Campos

EXTRAÇÃO DE CARACTERÍSTICAS EM SNORNAS USANDO MODELOS MATEMÁTICOS

Trabalho apresentado ao Curso de Bacharelado em Ciência da Computação do Instituto Federal de Educação, Ciência e Tecnologia de Brasília, campus Taguatinga, como requisito parcial para obtenção do grau de Bacharel em Bacharelado em Ciência da Computação.

Orientador: João Victor de Araujo Oliveira

BRASÍLIA
2023

Dedico este trabalho à todas as pessoas que me incentivaram, apoiaram e forneceram todo o suporte necessário para a construção da tese.

Agradecimentos

Agradeço à minha família por sempre ter me apoiado a estudar, principalmente aos meus pais que me nunca deixaram de prestar o suporte necessário durante a minha jornada. Ao Instituto Federal de Brasília (IFB) em conjunto com os docentes por terem consolidado o conhecimento necessário para chegar a esta etapa. Em especial ao orientador João Vitor, por ter me guiado e prestado assistência desde o início da realização do trabalho. E, claro, aos amigos e colegas que criei nessa jornada os quais convivi por longos anos e que me ajudaram a crescer em todos os aspectos.

Não há paixão a ser encontrada apostando pequeno - em se contentar com uma vida que é menor do que aquela que você é capaz de viver.

—NELSON MANDELA

Resumo

O número de sequências biológicas disponíveis aumentou significativamente nos últimos anos devido a várias descobertas científicas sobre o código genético que compõe os seres vivos, criando um enorme volume de dados. Por consequência, novos métodos computacionais foram moldados para analisar e extrair informações dessas sequências genéticas. Os métodos de aprendizado de máquina (AM) têm mostrado ampla aplicabilidade em bioinformática e demonstrou ser imprescendível para a extração de informações úteis das estruturas secundárias dos genomas ao aperfeiçoar suas técnicas com base no arquétipo matemático em contraste com o modelo padrão biológico de análise. Diante disso, este trabalho visa analisar os modelos matemáticos de extração de características, principalmente as técnicas de extração que demonstraram ser eficientes na classificação de snoRNAs *C/D box* em organismos vertebrados e invertebrados com um F-score de 98% e na classificação de snoRNAs *H/ACA box* com um *F-score* de 95%. Os algoritmos como os da Transformação Numérica de *Fourier* e de Redes Complexas atingiram uma taxa maior que 90% na classificação de snoRNAs *C/D box* e *H/ACA box* em sequências genéticas de *Homo Sapiens, Platypus, Gallus gallus, Nematodes, Drosophila* e *Leishmania* demonstrando ser promissor nas moléculas de RNAs não-codificadores (ncRNA) da classe de snoRNAs.

Palavras-chave: RNAs não codificadores, snoRNAs, Aprendizagem de Máquina, Modelos matemáticos de extração de características, classificação de sequências biológicas, *C/D box*, *H/ACA box*, *Random Forest*

Abstract

The number of biological sequences available has increased significantly in recent years due to several scientific discoveries about the genetic code that composes living beings, creating a huge volume of data. Consequently, new computational methods were shaped to analyze and extract information from these genetic sequences. The learning methods (AM) have shown wide applicability in bioinformatics and proven to be essential for the selection of useful information from the secondary structures of genomes by perfecting his techniques based on the mathematical archetype in contrast to the model biological standard of analysis. Therefore, this work aims to analyze the mathematical models for feature extraction, mainly extraction techniques that were verified efficient in classifying C/D box snoRNAs in vertebrate and invertebrate organisms with an F-score of 98% and in classifier snoRNAs as H/ACA box with an F-score of 95%. Algorithms such as Fourier Numerical Transformation and Complex Networks reached a greater than 90% rate in classifying C/D box and H/ACA box snoRNAs in genetics sequences of Homo Sapiens, Platypus, Gallus gallus, Nematodes, Drosophila and Leishmania proving to be promising in non-coding RNA (ncRNA) molecules of the class of snoRNAs.

Keywords: non-coding RNAs, snoRNAs, Machine Learning, mathematical models for feature extraction, biological sequences classification, C/D box, H/ACA box, Random Forest

Lista de Figuras

Lista de Tabelas

Lista de Acrônimos

Sumário

1

Introdução

A expansão do Aprendizado de Máquina (AM) e de técnicas biológicas para a predição de estruturas protéicas e genômicas e também para o diagnóstico de doenças, trouxe um resultado significado no que tange a identificação de padrões e características em RNAs não-codificadores *(ncRNAs)*. De acordo com BONIDIA et al. (2021a), existem aplicações modernas que extraem propriedades biológicas relevantes para o estudo dessas moléculas como quadro de leitura aberto (*ORF*), o uso da frequência de *triplets* (nucleotídeos "trigêmeos"adjacentes) e a porcentagem de conteúdo *GC*. A aplicabilidade de cunho biológico, apesar de expressivo, dificilmente tem reutilização ou adaptação para problemas específicos tal como classificar as classes de RNAs não-codificadores (ncRNAs).

Um exemplo disso é a classe de pequenos RNAs nucleolares (snoRNAs) que podem ser divididos em duas classes: *C/D box* e *H/ACA box*. Em uma sequência de ncRNA, através da extração de características da estrutura secundária, em conjunto com técnicas de aprendizado supervisionado, auxiliam na identificação das classes *C/D box* e *H/ACA box snoRNAs*, como visto na dissertação de ARAUJO (2017).

A construção de um modelo preditivo devido às limitações dos experimentos manuais no laboratório a fim de otimizar o desempenho dos modelos atuais de aprendizado de máquina também inclui uma representação matemática das sequências biológicas por meio do mapeamento numérico e a transformação de Fourier. A adoção de uma abordagem matemática no contexto de *ncRNAs* demonstrou ser promissora nos experimentos de BONIDIA et al. (2021a) ao comparar a sua eficiência com os algoritmos particulamente de natureza biológica computacional.

O pacote *MathFeature*, proposto por BONIDIA et al. (2021b), contém 37 *features* descritivas para sequencias biologicas. Dentre estas 37, 20 são baseadas em uma analise matemática incluindo tanto a transformação de *Fourier* quanto o mapeamento numérico, mas também a entropia, grafos, redes complexas e CGR (*Chaos game representation*) em sua composição. Os casos de estudo alcançaram resultados experimentais significativos.

1.1 Formulação do problema

A extração de características busca gerar um vetor de características para uma determinada estrutura baseado no treinamento intensivo do modelo. A busca por técnicas de AM capazes de identificar as características de estruturas secundárias de RNAs tornou-se fundamental ao longo dos últimos anos devido a grande quantidade de dados sobre o conteúdo genético.

Os métodos tradicionais de extração de características do AM nem sempre conseguem determinar um modelo eficaz que consiga evitar a perda de informações da estrutura, um bom exemplo disso é para as classes de snoRNAs. Para ilustrar a ideia, o estudo de BONIDIA et al. (2021a), é relatado que a técnica *ORF*, cuja é bastante aplicada no meio biológico para leitura sequencial do códon, retornou uma pontuação inferior a 0,009 para classificar o RNA circular de outros tipos de lncRNAs.

Além desse fato, os autores de BONIDIA et al. (2021b) mostraram que a eficiência de algoritmos para classificação de classes de lncRNAs demonstraram amplo proveito nos estudos de caso obtendo um desempenho entre 0.6350–0.9897 de eficácia na fase de avaliação, o que é extremamente vantajoso para a classificação de lncRNAs.

Diante disso, a questão norteadora do trabalho é assumida na hipótese a seguir:

- **Hipótese** - Os métodos matemáticos, apesar de generalistas, são bons o suficiente como os biológicos para classificar as duas classes de snoRNAs: *H/ACA box* e *C/D box*.

Pretende-se analisar e fundamentalizar a hipótese baseado nos resultados obtidos de experimentos em casos de testes.

1.2 Objetivos

1.2.1 Objetivos gerais

Este trabalho tem como objetivo geral a análise de modelos matemáticos de extração de características para as classes de snoRNAs *C/D box* e *H/ACA box*.

1.2.2 Objetivos específicos

1. Realizar a coleta e tratamento de dados de snoRNAs para a criação do conjunto de dados de treinamento e teste;

2. Usar um algoritmo de extração de *features* com abordagem matemática como a transformação de Fourier, mapeamento numérico, entropia (Shannon e Tsallis), redes complexas, EDeN e/ou etc;

3. Extrair as *features* a partir de modelos matemáticos de ambas as classes de snoRNAs (*H/ACA box* e *C/D box*);

4. Avaliar o desempenho de diferentes técnicas de extração de características no algoritmo *Random Forest*;

5. Avaliar o performance dos melhores modelos gerados em sequências identificadas na literatura;

Os capítulos estão divididos em torno do referencial teórico, revisão da literatura, capítulo de projeto, resultados e conclusão. O referencial teórico contextualizará, na seção de Bioinformática, o que são moléculas de RNA, o mecanismo molecular e sua síntese proteíca e explicará sobre os tipos de ncRNAs, dando ênfase na família de snoRNAs.

Ainda no referencial teórico, haverá uma seção específica para descrever o conceito de AM e os algoritmos de aprendizado supervisionado ou não-supervisionado. Será destacado os algoritmos SVM, *CNN*, EDeN e *Random Forest* e o seu funcionamento em torno dos diversos problemas de classificação e regressão.

Em sequência, os métodos de extração de características de natureza matemática serão abordados como as transformações numéricas de Fourier, as entropias de Tsallis e Shannon e as redes complexas. Cada algoritmo tem sua particularidade e importância de atuação na identificação de atributos das sequências genética, o que será enfatizado nas subseções deste capítulo.

Após a definição dos algoritmos de extração, será introduzido o conceito de sobre-ajuste (*overfitting*), um problema recorrente na classificação que deve ser evitado a todo custo para que não se tenha resultados adulterados.

Finalmente, a técnica de validação cruzada terá seu destaque pois é um dos métodos aplicáveis que previne o sobre-ajuste de dados nas etapas de treinamento e testes. Dessa forma, com os resultados do classificador autênticos, a última seção explica as métricas de avaliação que serão consumidas para apreciação do modelo.

As revisões de literatura explicam a construção do estado de conhecimento, as questões de pesquisa que nortearam o trabalho, as estratégias de busca em torno dos bancos de dados e repositórios, os critérios de inclusão e exclusão de artigos literários e por fim, a análise e discussão das literaturas que mais tiveram notoriedade como referência a esta monografia.

No capítulo de projeto as etapas de AM serão explicadas detalhadamente conforme foi empregado no trabalho. Desde a coleta de dados, a etapa de pré-processamento de dados, extração de características, até treinamento e testes considerando os estudos de caso definidos em torno da classificação de snoRNAs.

Até que enfim o capítulo de resultados demonstra a eficiência do classificador, as estatísticas minuciosas de cada método de extração na fase de testes, o comparativo entre o software snoReport 2.0 da literatura de ARAUJO (2017) nas sequências de organismos vertebrados e invertebrados das classes snoRNAs comentados nas literaturas previstas.

2

Referencial teórico

As próximas seções abordarão os principais conceitos sobre cada temática da bioinformática apontando o campo de pesquisa para os ncRNAs do tipo snoRNAs.

Na primeira seção será discutido sobre a importância da bioinformática na área científica apresentando o objeto de pesquisa desde os ácidos nucleicos e sua sintese proteica até os ncRNAs e por fim os snoRNAs da classe C/D e H/ACA.

Na segunda seção será retratado a definição de apredizagem de máquina explicando as suas características, metodologia, fluxo de trabalho e os principais algoritmos de classificação e regressão.

A terceira seção será responsável em explicar os algoritmos usados neste trabalho de extração de característica de natureza matemática (Transformação de Fourier com o mapeamento numérico Real e *Z-Curve*, Entropia de Shannon e Tsallis e Redes Complexas)

A quarta seção irá explicar sobre um dos problemas mais comuns no espectro de AM chamado *overfitting*. Esta seção irá detalhar a relação causa e consequência do sobre-ajuste e mostrará as possíveis tomadas de decisão que irão evitar e prevenir que esta situação ocorra no seu modelo.

A penúltima seção explicará a importância da validação cruzada na fase de treinamento e de testes na aprimoração do modelo preditivo, a ideia é constatar que o particionamento do conjunto em pequenas partes influenciará no resultado final do classificador.

Finalmente, na última seção, as métricas de avaliação serão elucidadas. Cada métrica avalia a acurácia do classificador e em sua maioria utilizam a matriz de confusão como base de cálculo estatístico. A partir de seus resultados, o modelo preditivo será ponderado.

2.1 Bioinformática

A bioinformática é a área de pesquisa que mescla as ciências biológicas e computacionais para o gerenciamento computacional de todos os tipos de informações biológicas moleculares, lidando com a estrutura e os aspectos funcionais dos genes e proteínas. Têm como objetivo desenvolver técnicas modernas para identificação de características do objeto de análise com o intuito de expandir a produção de farmacológicos, o descobrimento de vacinas, cura para doenças,

dentre outros. Os estudos sobre o sequenciamento de moléculas genéticas e a análise sistemática de cadeias protéicas dos acidos nucleicos (DNAs e RNAs) evidenciaram que existe uma classe de pequenas moléculas de RNAs que desempenham um papel importante nos processos celulares RANA; VAISLA (2012).

2.1.1 Ácidos nucleicos

De acordo com SETUBAL; MEIDANIS (1997), ao que tange sobre ácidos nucleicos, são moléculas formadas por nucleotideos responsáveis por armazenar e transmitir as informações genéticas necessárias para a produção de proteínas, bem como a transcrição do conteúdo a partir do processo de reprodução celular. Os seres vivos, em sua composição, contém dois tipos importantes de ácidos nucleicos: o DNA e o RNA. O DNA (ácido desoxirribonucleico) detém a função de armazenar e transmitir essas informações enquanto o RNA (ácido ribonucleico) transcreve-as para formar a síntese protéica.

Na biologia molecular, dois nucleotídeos em fitas complementares de DNA ou RNA que estão conectados por ligações de hidrogênio são chamados de par de bases nitrogenadas. No pareamento de bases Watson-Crick canônico no DNA, a adenina (A) forma um par de bases com a timina (T) usando duas ligações de hidrogênio, e a guanina (G) forma um par de bases com a citosina (C) usando três ligações de hidrogênio. No pareamento de bases Watson-Crick canônico no RNA, a timina é substituída por uracila (U) WATSON–CRICK PAIRING (2011). A figura 2.1 demonstra como funciona as ligações entre as bases nitrogenadas.

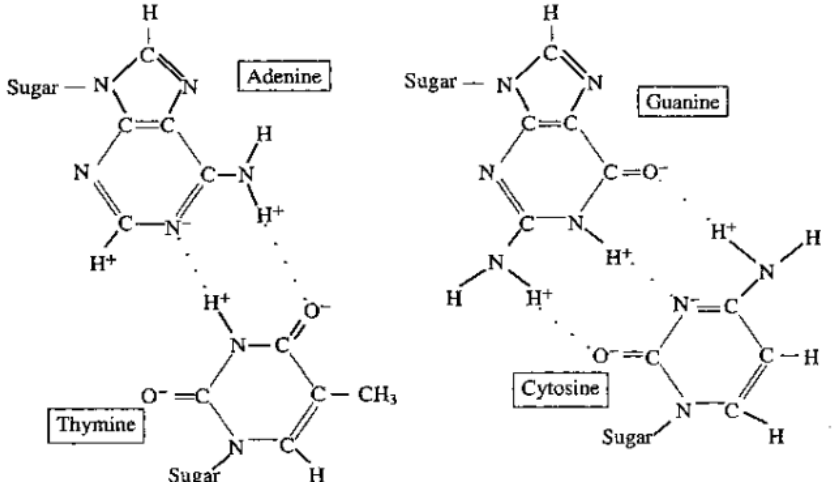


Figura 2.1: Extraído de SETUBAL; MEIDANIS (1997) para representar as ligações entre as bases nitrogenadas.

2.1.2 Mecanismo molecular e síntese protéica

O mecanismo celular reconhece o início de um gene ou agrupamento de genes graças ao promotor, que é uma região antes de cada gene no DNA que serve de indicação ao mecanismo

celular que um gene está à frente. Tendo reconhecido o início de um gene ou agrupamento de genes, uma cópia do gene é feita em uma molécula de RNA. Este RNA resultante é o RNA mensageiro (mRNA) e terá exatamente a mesma sequência que uma das fitas do gene, mas substituindo a base nitrogenada U por T. Esse processo é chamado de transcrição. O mRNA resultante será então usado em estruturas celulares chamadas ribossomos para fabricar uma proteína.

Como o RNA é de fita simples e o DNA é de fita dupla, o mRNA produzido é idêntico em sequência a apenas uma das fitas gênicas, sendo complementar à outra fita, tendo em mente que T é substituído por U no RNA. A fita que se parece com o produto de mRNA é chamado de fita *antisense* ou codificadora, e a outra é a fita *sense* ou anticodificação ou então fita molde. A fita molde é a que realmente é transcrita, pois o mRNA é composto pela união de ribonucleotídeos complementares a esta fita.

A proteina é uma macromolécula formada por uma cadeia de aminoácidos pareadas por uma ligações peptídicas que conectam um átomo de carbono pertencente à carboxila a um ou mais átomos de nitrogênio. Ao efetuar a ligação, uma molécula de água é liberada porque o oxigênio e o hidrogênio da carboxila se une a um hidrogênio do grupo de amina. Assim, o que realmente encontramos dentro de uma cadeia polipeptídica é um resíduo do aminoácido original. As proteínas típicas contêm cerca de 300 resíduos, mas existem proteínas com apenas 100 ou com até 5.000 resíduos SETUBAL; MEIDANIS (1997). A figura 2.2 apresenta alguns aminoácidos bastante importantes da cadeia polipeptídica.

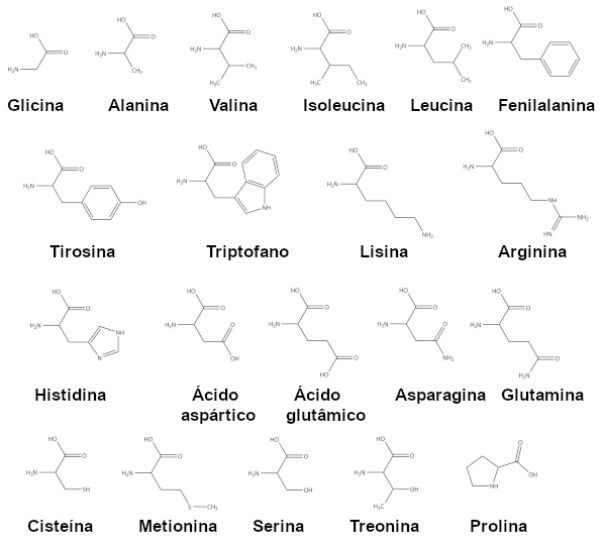


Figura 2.2: Moléculas de aminoácidos conhecidas. SANTOS (2019)

Portanto, para identificar uma proteína, é necessário decodificar cada aminoácido que ela contém. E isso é precisamente o que o DNA em um gene faz, usando triplas de nucleotídeos para especificar cada aminoácido. Cada tripla de nucleotídeos é chamado de códon. As triplas de nucleotídeos são dados usando bases de RNA em vez de bases de DNA, a razão é que são as moléculas de RNA que fornecem a ligação entre o DNA e a síntese protéica em um processo

chamado de tradução. A conexão que designa a síntese protéica é feita entre um códon e o aminoácido que este códon codifica. Cada molécula de RNA transportador (tRNA) possui, de um lado, uma conformação que possui alta afinidade por um códon específico e, do outro lado, uma conformação que liga-se facilmente ao aminoácido correspondente. À medida que o RNA mensageiro passa o interior do ribossomo, um tRNA correspondente ao códon atual se liga a ele, trazendo o aminoácido RANA; VAISLA (2012).

Um aspecto do processo de transcrição importante é o conceito de quadro da leitura. Um quadro de leitura aberto, ou ORF, em uma sequência de DNA é um trecho contíguo dessa sequência começando no códon inicial, tendo um número inteiro de códons (seu comprimento é um múltiplo de três) tal que nenhum de seus códons seja um codão de terminação (uma tripla de nucleótidos que sinaliza a terminação da tradução). Uma das três formas possíveis de agrupar bases para formam códons em uma sequência de DNA ou RNA. Por exemplo, a sequência **TAATCGAATGGGC** pode ser decodificada tomando como códons TAA, TCG, AAT, GGG, deixando de fora o último C. Outro quadro de leitura seria ignorar o primeiro T e obter os códons AAT, CGA, ATG, GGC. Ainda outro quadro de leitura produziria os códons ATC, GAA, TGG, deixando de fora duas bases no início (TA) e duas bases no final (GC) PAYNE (2017). Visualmente, a imagem 2.3 apresenta a estrutura de um RNA do vírus *SARS* coronavírus, mostrando que é possível trocar as bases nitrogenadas se a transcrição para códon não afetar as suas ligações peptídeas.

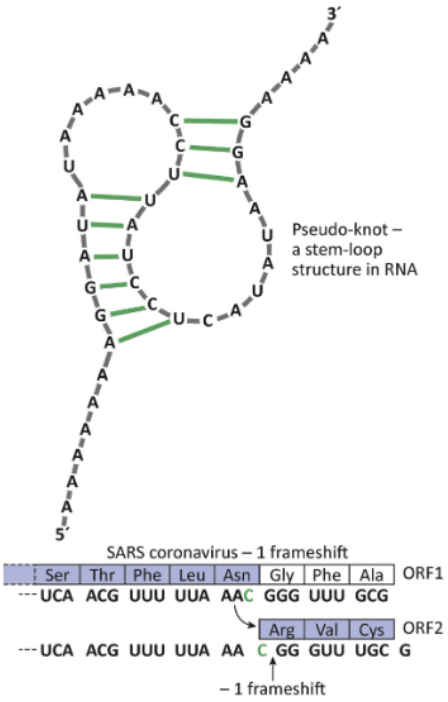


Figura 2.3: Um pseudo-nó de RNA direcionando a estrutura ribossômica na síntese protéica. Extraído de PAYNE (2017)

2.1.3 ncRNAs

As regiões de proteínas não-codificadas abrangem 98% do genoma humano e caracterizam por ser uma região a qual os RNAs detectados não são codificados a partir da síntese protéica. Os RNAs não traduzidos em proteínas foram nomeados RNAs não-codificadores (ncRNA) e foram considerados, a princípio, como um ruído ou subprodutos do fluxo de informação genética do DNA à proteína. A falta de dados científicos atribuindo funções para a maioria das regiões não codificantes do genoma reforça a ideia de que essa maioria pode realmente ser descartável. No entanto, hoje, é conhecido que os ncRNAs estão envolvidos em várias atividades celulares, como silenciamento de genes, replicação, regulação da expressão gênica, transcrição, estabilidade cromossômica, estabilidade de proteínas, translocação e localização, modificação de RNA, processamento e estabilidade CHAKRAVORTY (2022).

A predição de estruturas conservadas é um fator preponderante para descobrir e caracterizar assinaturas para uma família de RNA específica. Ao tratar sobre ncRNAs, a sua identificação está estritamente ligada à sua estrutura terciária e, como a estrutura terciária é determinada pela estrutura secundária, esta última é usada como uma aproximação no estudo de funções em ncRNAs, para ter uma melhor noção de como é a estrutura de um ncRNA, a figura 2.4 mostra sua composição em forma de grafo. Se a função de um único RNA ou de uma família não for conhecida, pode-se inferir comparando a estrutura de RNA (ou consenso no caso de uma família) com um banco de dados de assinaturas estruturais secundárias. A comparação estrutural pode também ser usada para detectar a ocorrência de diferentes estruturas estáveis para a mesma molécula (o que pode indicar uma possível mudança na estrutura secundária impactando diretamente na sua função) para prever e comparar as mutações em uma sequência de RNA GUSIC; PROKISCH (2020)

Figura 2.4: Estrutura secundária do icd-II ncRNA. Extraída de GUSIC; PROKISCH (2020)

A designação de famílias de ncRNAs a partir da comparação estrutural secundária de sequências de ncRNAs, procurando RNAs homólogos a um candidato específico ou que pertence a uma família de candidatos continua sendo um problema específico da classificação de ncRNAs. Em tese, a pesquisa de ncRNA envolve três tipos principais de problemas de reconhecimento de padrões em ncRNas, segundo LIMA; PORTILLO (2007), são eles:

- Predição da estrutura secundária: O número de estruturas secundárias possíveis cresce exponencialmente com o comprimento da sequência. A questão é como buscar uma estrutura neste espaço de solução exponencial para escolher a melhor estrutura. Quando a estrutura secundária de apenas uma sequência de RNA precisa ser prevista, apenas métodos iniciais podem ser usado. Se um conjunto de RNAs homólogos estiver disponível, métodos comparativos podem prever a estrutura de consenso com mais precisão.

- Comparação de estrutura secundária: A comparação de estrutura calcula a diferença entre duas estruturas. A medição é feita calculando a diferença de uma distância de edição entre essas duas estruturas. A distância de edição depende de quantas operações de edição são necessárias para transformar uma das estruturas em outra considerando o custo de cada tipo de operação de edição. O cálculo da distância de edição está diretamente relacionada à forma como as estruturas são representadas e em que nível de resolução a comparação é realizada. Três maneiras comuns de representar estruturas são árvores, cadeias de colchetes e gráficos genéricos. Os níveis de resolução variam de pares de bases para padrões estruturais como hélices, *loops* e *multi-loops*.

- Identificação de ncRNAs: A detecção computacional direcionados a famílias de ncRNAs usam o máximo possível de peculiaridades. A criação de programas mais gerais que possam ser treinados para identificar características de uma família específica ou mesmo de uma única sequência de entrada é um caminho possível para identificação de famílias. Ainda assim, é desejável procurar novas famílias de genes, o que torna um problema para os programas de classificação geral de famílias conhecidas.

2.1.4 snoRNAs

Os snoRNAs são uma das mais antigas e numerosas famílias de RNAs não codificantes (ncRNAs), estão amplamente presentes nos nucléolos das células eucarióticas e têm um cumprimento de 60–300 nt. A principal função dos snoRNAs é guiar a modificação de RNA ribossomal (rRNA) específica do local. Em contraste, sua organização genômica e estratégias de expressão são as mais variadas. Aparentemente, as unidades de codificação de snoRNA adotaram, no curso da evolução, todas as formas possíveis de serem transcritas, proporcionando assim um paradigma único de flexibilidade de expressão gênica. DIECI et al. (2009)

Os snoRNAs são codificados principalmente por regiões intrônicas de genes codificadores de proteínas e não codificadores de proteínas. Normalmente, podem ser classificados em três grupos: *H/ACA box*, *C/D box* e RNAs cajal pequenos (scaRNAs). Para HUANG et al. (2022), os dois primeiros tipos de snoRNAs participam do processamento de rRNA adicionando modificações de 2-O-metilação e pseudouridilatação às moléculas de rRNA, respectivamente. No entanto, um tipo de snoRNAs estão localizados em corpos de Cajal (CBs), eles são chamados scaRNAs. Eles também seguem a classificação C/D-H/ACA, mas alguns scaRNAs contêm estruturas C/D e H/ACA. *C/D box* snoRNAs se ligam a quatro proteínas essenciais – Nop1p, Nop56p, Nop58p e Snu13p — para gerar pequenas ribonucleoproteínas nucleolar (snoRNPs). Da mesma forma, os snoRNAs *H/ACA box* formam snoRNPs funcionais ligando-se a Cbf5p, Gar1p, Nhp2p e Nop10p.

O comprimento dos snoRNAs da *C/D box* eucariótica geralmente varia de 70 a 120 nt, como pode ser relatado pela figura 2.5. Esses snoRNAs contêm duas sequências conservadas: a *C box e a D box*. A *C box* consiste nos nucleotídeos **RUGAUGA**, que estão localizados na extremidade 5' da molécula de snoRNA. Em contraste, a *D box* está localizada na extremidade 3' e consiste nos nucleotídeos **CUGA**. Juntos, esses elementos dependem do par de bases para dobrar em uma estrutura chamada *kink-turn*. Essa estrutura é reconhecida pelo Snu13p, que então recruta Nop1p (também chamado fibrilarina [FBL]), Nop58p e Nop56p para modificação de 2'-O-metilação DIECI et al. (2009).

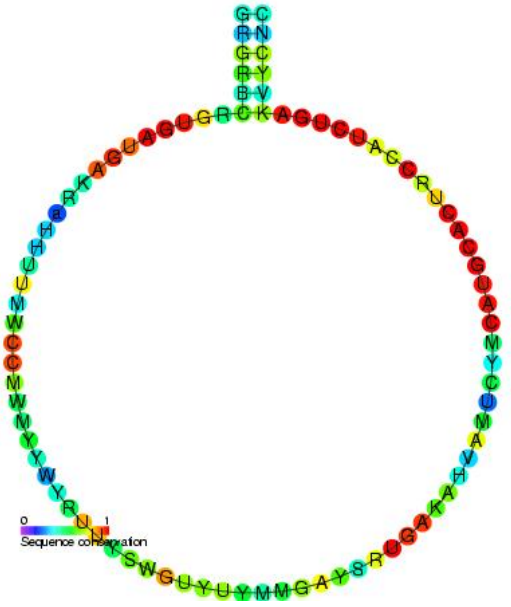


Figura 2.5: Estrutura secundária do SNORD33, que pertence ao grupo *C/D box*.

Os snoRNAs *H/ACA box* contêm a região chamada de bolsas de pseudouridilatação em que há resíduos de uridina no substrato RNA isomerizados. *H/ACA box* snoRNPs se ligam a Cbf5p, Nop10p, Gar1p e Nhp2p, entre os quais Cbf5p atua como a proteína catalítica envolvida na pseudouridilatação. Os snoRNAs eucarióticos *H/ACA box* contêm duas sequências: a *H box* e a *ACA box*, localizadas abaixo do primeiro e segundo *hairpin*, respectivamente. DIECI et al.

(2009). Um exemplo de snoRNA *H/ACA box* é visto na imagem 2.6

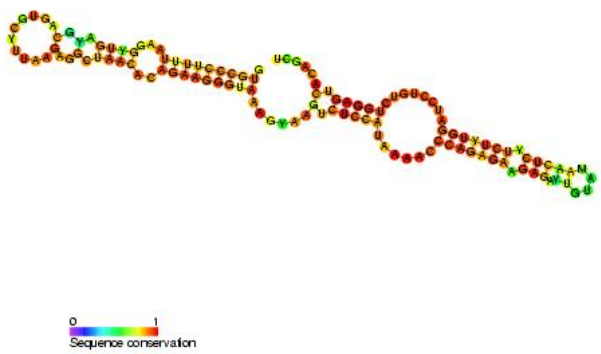


Figura 2.6: Estrutura secundária do SNORA26, que pertence ao grupo *H/ACA box*.

Os métodos de AM são frequentemente usados na identificação e classificação de diferentes famílias de ncRNAs como snoRNAs. Uma avaliação sistemática de AM nos snoRNAs requer um aprendizado supervisionado em torno das classes para que possa extrair os atributos, chamados de *features*, significativos do material genético. Essas *features* serão consumidas por um classificador, em outras palavras, por um algoritmo de AM, que será responsável de predizer, avaliar e validar o modelo preditivo encontrado. Para aprofundar nesses métodos de identificação e classificação, é necessário compreender como funciona o *workflow* do Machine Learning (ML), os algoritmos de aprendizado supervisionado, os algoritmos de extração de características (em priori os de gênero matemático), como funciona a avaliação métrica de um classificador e as possíveis consequências que certas escolhas de parâmetro para o algoritmo podem induzir.

2.2 *Machine learning*

De acordo com ALLALI et al. (2021), o AM é um ramo da inteligência artificial que envolve a autoaprendizagem do computador para executar tarefas. A seleção de características no aprendizado de máquina têm um papel significativo no desempenho dos modelos de previsão. É durante a seleção que a redundância e ruídos são identificados, a remoção do sobre-ajuste é aplicado, o que implica diretamente no aumento da velocidade de cálculo. Esta etapa crucial é capaz de definir as características discriminantes do objeto de estudo analisado. Para entender melhor, MITCHELL (1997) explica o funcionamento do AM sugerindo uma esquematização de um programa de computador o qual aprende a partir de uma experiência E através de alguma classe de tarefas T e uma medida de desempenho P. Vale ressaltar que sua performance para a tarefa T, medida em P, é aprimorada com a experiência E. Por exemplo, considerando a

aplicabilidade do objeto de estudo da tese, considere que um programa deve classificar uma determinada sequência de código genético e precisa classificá-lo como um RNA, portanto:

O problema pode ser traduzido para:

- Tarefa T: Classificar o código genético em DNA ou RNA
- Medida de desempenho P: Percentual de sequências de RNAs classificadas corretamente
- Experiência E: Um banco de dados de sequências conhecidas de DNAs e de sequências de RNAs

Dado um conjunto de dados, o algoritmo fará o treinamento a partir das características (*features*) predefinidas que descrevem o RNA. A hipótese inicial é que haja uma função f que consiga ser aplicável à um grupo X de código genético que o caracterize como um RNA. O algoritmo não retorna uma solução exata, logo, a correlação do resultado é baseado na margem de erro da função heurística. Todavia, a intenção do aprendizado de máquina é que torne a máquina consistente ao armazenar a "experiência"ou informação advindas do banco de dados; quanto mais características estiver disponível para definir o caso de estudo, melhor. A determinação da acurácia é feita pelo valor resultante do *F-score*, em termos estatísticos, é a medida de precisão de um teste, portanto, a escolha do conjunto de dados bem como a preferência do algoritmo impactam diretamente na sua eficácia HENNIG et al. (2021).

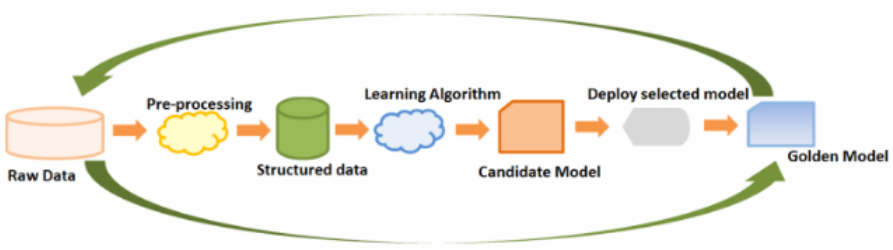


Figura 2.7: Fluxo de trabalho do aprendizado de máquina.

A figura 2.7 mostra um simples fluxo de trabalho do aprendizado de máquina, inclusive, muito utilizado na mineração de dados: na primeira etapa recebe o dado bruto e em seguida passa para etapa de pré-processamento que filtrará os dados e os limpará deixando-o estruturado, em sequência o algoritmo escolhido é executado e retornará o modelo candidado para o treinamento em questão, a verificação do *F-score* é feito na fase seguinte e o "modelo de ouro", isto é, o grupo classificado que obteve a maior acurácia é encontrado. HENNIG et al. (2021) O aprendizado de máquina é dividido em três grupos: aprendizado supervisionado, não supervisionado e reforçado. O aprendizado híbrido, por sua vez, combina ambos métodos: supervisionado e não supervisionado.

No estudo de HENNIG et al. (2021), define-se que o aprendizado supervisionado mapeia uma entrada para uma saída com base em um conjunto de dados conhecidos, a saída é uma classe (no caso de classificação) ou um valor (em regressão linear). Na aprendizagem não supervisionada, os algoritmos construem modelos capazes de descrever os dados e as relações encontradas sem o uso de rótulos, além de incluir a divisão de dados em grupos (no caso de agrupamento) e resume a distribuição de dados em densidade estimativa. A aprendizagem por reforço envolve ações de aprendizagem em vez de classe, e a entrada é mapeada para ações com base no retorno, logo, é orientada a ação, onde são mantidas as ações que conterem maior recompensa. Para elucidar, as próximas seções abordarão alguns algoritmos de classificação importantes como *SVM*, *CNN* e *Explicit Decomposition with Neighborhoods* (EDeN), *Random Forest* para ilustrar alguns dos métodos disponíveis do aprendizado de máquina.

2.2.1 Etapas do *Machine Learning*

As etapas do *Machine Learning* podem ser divididas em 5 estágios segundo PANT (2019):

1. **Coleta de dados**: A máquina de aprendizado inicialmente aprende com os dados que são fornecidos à ela. A qualidade dos dados que alimenta a máquina determinará a precisão do modelo preditivo. Dados incorretos ou desatualizados terão resultados ou previsões erradas que não são relevantes. Portanto, a primeira etapa no processo de aprendizado de máquina é obter os dados autênticos para construção do conjunto positivo e negativo, podendo ser adquiridos de bancos de dados existentes ou de repositórios *online*, desde que seja de fonte confiável.

2. **Pré-processamento de dados**: Todos os dados do mundo real geralmente não estão bem estruturados, redundantes ou não tem informação descricionária. Para explorá-los no modelo de aprendizado de máquina, é inevitável que haja uma preparação, limpeza, no intuito de que fique mais claro e objetivo. Essa etapa é crucial no fluxo de trabalho do ML e também a que leva mais tempo. Os dados podem estar em qualquer formato: CSV, XML, JSON, etc. Após a conversão para um formato padrão, é preciso limpá-los. Sendo assim, é fundamental checar se a quantidade de dados para os conjuntos (positivo e negativo) estão balanceados, se as sequências genéticas detém tamanhos semelhantes, se o genoma contém apenas bases nitrogenadas, e etc.

3. **Extração de característica**: A extração de características refere-se ao processo de transformação de dados brutos em dados numéricos que podem ser processados, preservando as informações no conjunto de dados original. É uma etapa primordial, pois o algoritmo de aprendizado de máquina produz melhores resultados com valores contínuos e discretos do que diretamente com dados brutos. Escolher um algoritmo de extração capaz de transcrever os atributos de cada sequência genética é decisivo

acerca o resultado da predição: a eficácia da predição pode variar conforme a técnica escolhida de extração.

4. **Treinamento**: A próxima etapa no fluxo de trabalho de aprendizado de máquina é treinar o modelo. Um algoritmo de ML é empregado no conjunto de dados de treinamento com o objetivo de aprender e prever certos "comportamentos"baseado nos valores reais advindos da extração. Esses algoritmos podem se enquadrar em três grandes categorias: binário, classificação e regressão. Neste trabalho será usado o algoritmo de classificação.

5. **Testes**: Depois que o modelo é treinado exaustivamente, o próximo passo é testá-lo e validá-lo para garantir que eficaz. Usando o conjunto de dados de teste obtido na etapa 3, é feito a verificação da precisão do modelo obtido. O modelo pode ser treinado, alterado e aprimorado várias vezes até que os resultados sejam satisfatórios.

Todo o passo-a-passo pode ser descrito pela figura 2.8.

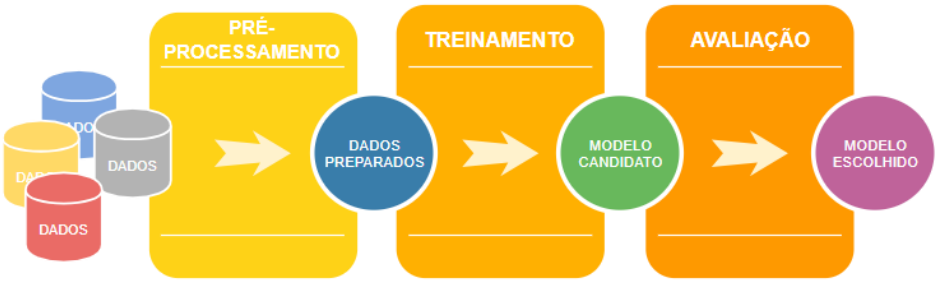


Figura 2.8: Fluxo de trabalho do ML; Imagem extraída de SCHADE (2018).

2.2.2 SVM

As Máquinas de Vetores de Suporte (do inglês, Support Vector Machines - SVMs) é um método não paramétrico que não é limitado pelo tamanho do conjunto de dados de treinamento. Essencialmente, o Máquina de vetores de suporte (SVM) gera modelos usados para classificação e regressão CHEN et al. (2015). Em ambos os casos, se o SVM não for capaz de criar os vetores de suporte, ele pode construir hiperplanos em uma dimensão alta no espaço euclidiano para que ele selecione aqueles com maior margem, relacionados aos dados de treinamento VIEIRA et al. (2017). Nessas circunstâncias, o modelo SVM tenta encontrar uma reta para distinguir os grupos da entrada do conjunto de dados, como ilustrado pela figura 2.9, excluso os casos em que as margens não podem ser criadas quando métodos simples de separação linear são usados para dados não-lineares.

Com a finalidade de resolver este obstáculo, o SVM usa funções do kernel para aumentar a dimensão do espaço, de modo que o conjunto de dados possa ser linearmente separável em

dimensões mais altas, como na figura 2.10, o qual o algoritmo divide o plano bidimensional de forma que separe os vetores de suporte em ambas extremidades da reta.

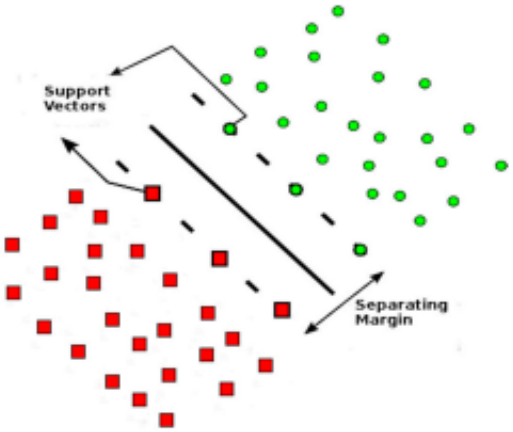


Figura 2.9: Exemplo de vetores de suporte de 2 dimensões adaptado por VIEIRA et al. (2017).

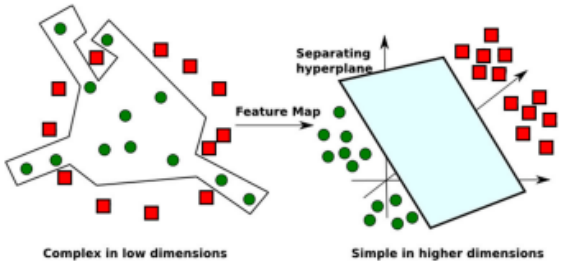


Figura 2.10: Dados separáveis não lineares em baixa dimensão, mapeados para uma dimensão mais alta; adaptado por VIEIRA et al. (2017).

Uma noção que é central para a construção do algoritmo de aprendizado do vetor de suporte é o *kernel* do produto interno entre um "vetor de suporte"xi e o vetor x extraído do espaço de entrada. Os vetores de suporte consistem em um pequeno subconjunto dos dados de treinamento extraídos pelo algoritmo

2.2.3 *CNN*

O trabalho de Hubel e Wiesel em 1962 sobre a descoberta de atividades elétricas de neurônios em gatos foi pioneira para o desenvolvimento do método de AM baseada em multicamadas de neurônios, chamada de rede convolucional neural (CNN) WURTZ (2009).Uma rede convolucional é um perceptron multicamada projetado especificamente para reconhecer formas bidimensionais com um alto grau de invariância à tradução, dimensionamento e distorção. Esta tarefa é aprendida de forma supervisionada por meio da rede cuja estrutura inclui as seguintes formas de restrições VIEIRA et al. (2017):

1. Extração de características - Cada neurônio recebe suas entradas sinápticas de um campo receptivo local na camada anterior, forçando-o a extrair características locais. Uma vez que um recurso foi extraído, sua localização exata torna-se menos importante, desde que sua posição em relação a outras características seja aproximadamente preservada.

2. Mapeamento de extrações - Cada camada computacional da rede é composta de vários mapas de características, com cada mapa de características na forma de um plano dentro do qual os neurônios individuais são restringidos a compartilhar o mesmo conjunto de pesos sinápticos. Esta segunda forma de restrição estrutural tem efeitos benéficos como a invariância de deslocamento e redução da quantidade de parâmetros livres recebidos pelos perceptrons.

3. Subamostragem - Cada camada convolucional é seguida por uma camada computacional que realiza média local e subamostragem, reduzindo a resolução do mapa de características. Esta operação tem o efeito de reduzir a sensibilidade da saída do mapa de características para deslocamentos e outras formas de distorção.

A preparação da rede convolucional neural pode ser expressa na imagem 2.11. Desde o estágio da entrada dos dados até a última etapa da rede, a simplificação da informação anterior é executada em cada camada (*pooling*) e sua utilidade é diminuir a quantidade de recursos agrupados ajudando a reduzir o número de parâmetros necessários nas camadas posteriores.

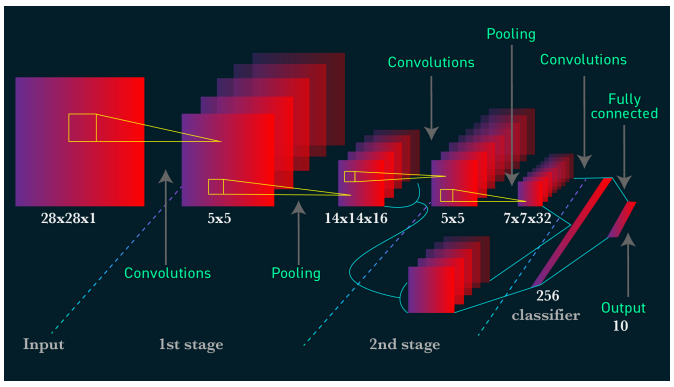


Figura 2.11: Representação da rede convolucional neural no processamento de imagens (I) entrada de dados, (II) primeiro estágio, (III) segundo estágio, (IV) classificador de 256 pixels, (V) saída com 10 pixels totalmente conectada; Imagem extraída de MALADKAR (2018).

A essência da rede neural convolucional é baseada na retropropagação que consiste na atualização contínua dos pesos para que os neurônios com a maior taxa de erro/perca sejam minimizados garantindo a consistência do modelo e a acurácia do método. A retropropagação funciona da seguinte forma: ao adentrarmos ao mapeamento de extrações, é necessário atualizar o peso de cada sináptico (ligação entre dois neurônios) de forma que aplique em todos os neurônios das camadas anteriores e para que isso seja implementado, é necessário a função de perca e a de

hipótese, elas guiarão o modelo pela a rede inteira até que chegamos a uma camada final que seja utilizável AL-MASRI (2019).

Portanto, conforme o mapeamento de extrações avança, os pesos tendem a diminuir pois são cálculos a partir da derivada parcial das funções. Consequentemente, o último nó da rede armazenará o total de perda do modelo que será usado para a avaliação do resultado, percebe-se, então, que a rede convolucional vai aprendendo a classificar o modelo por treinamento extraindo suas próprias características esporadicamente.

2.2.4 EDeN

Explicit Decomposition with Neighborhoods (EDeN) é um *kernel* decomposicional de grafos baseado no *Neighborhood Subgraph Pairwise Distance Kernel* (NSPDK) que produz um subgrafo da estrutura secundária do snoRNAs e produz um conjunto explícito de *features* usado para algoritmos de aprendizagem de máquina supervisionados, não-supervisionados ou híbridos de maneira escalável.

A decomposição de uma sequência genômica em partes de objeto pretende conceber um kernel local válido entre as subpartes para que seja obtido uma função de similaridade capaz de decompor exponencialmente se existir um método que enumere os kernels em tempo polinomial recorrendo a programação dinâmica para tal ato. À medida que a dimensão do espaço de características torna-se maior, há uma probabilidade de que uma fração das dimensões não serão correlacionadas com a função de similaridade. Como consequência, mesmo usando algoritmos de classificação com margem alta, tornam-se obsoletos para determinar uma boa generalização do modelo. COSTA; GRAVE (2010). A execução do EDeN é exemplificado pela imagem 2.12.

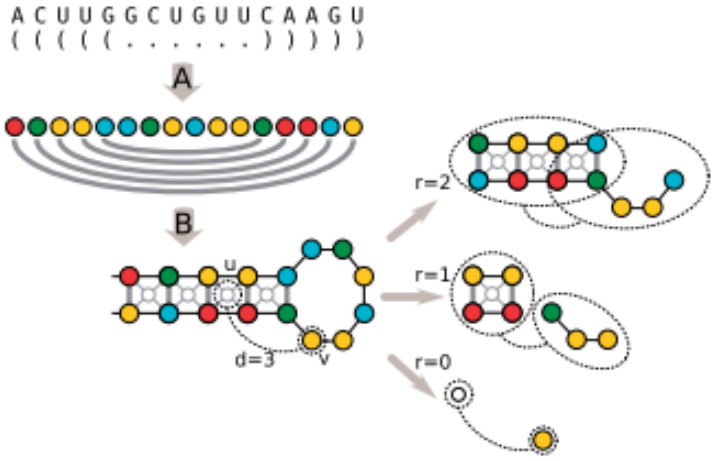


Figura 2.12: Codificação da estrutura secundária de RNA e as *features* do kernel do grafo; Imagem extraída de HEYNE et al. (2012).

Para entender melhor como funciona a tradução da sequência para um grafo, o passo-a-passo da codificação da estrutura secundária é: **(A)** A codificação do grafo preserva a informação

do nucleotídeo (rótulos do vértice) e os pares de bases (rótulos da borda), aqui representados com cores diferentes. **(B)** Vértices adicionais são inseridos para induzir *features* relacionados ao empilhamento quádruplo de pares de base (vértices finos de cor cinza no centro de cada empilhamento de pares). Na parte da direita exemplo de *features* induzidas pelo kernel do grafo *NSPDK* para um par de vértices **u,v** na distância 3 com raio 0,1,2. Os grafos de vizinhança são encerrados em trilhas tracejadas.

Em uma conotação matemática, proposto por HEYNE et al. (2012), dado um grafo $G = (V, E)$, em que V é o conjunto de vértices e o E é o conjunto de arestas. Sendo a distância de dois vértices u, v denotada por $D(u, v)$ do menor caminho entre eles e o raio r da região do subgrafo induzido é o conjunto de vértices a uma distância d menor ou igual a r de v. Considere que $N_{v,r}$ (G) denota o subgrafo de vizinhança, ou seja, o subgrafo de G enraizado em v induzido pelo conjunto de vértices. A relação de pares de vizinhança $R_{r,d}$ é definida como válida quando a distância entre as raízes de dois subgrafos de vizinhança de raio r é exatamente igual a d. O kernel de decomposição, portanto, $k_{r,d}$ na relação $R_{r,d}$ em um NSPDK pode ser definido da seguinte forma:

$$K(G, G^{'}) = \sum_{r}^{r^*} \sum_{d}^{d^*} k_{r,d}(G, G^{'}) \tag{2.1}$$

Isto significa que o NSPDK decompõe o grafo em pares de subgrafos vizinhos limitando a soma de $k_{r,d}$ kernels a cada iteração para todos os valores crescentes do parâmetro de raio r e distância d até um valor máximo dado r^* e d^* respectivamente.

2.2.5 *Random Forest*

O algoritmo *Random Forest* de AM é um conceito baseado em uma estrutura construída em árvores a partir da partição recursiva do conjunto de dados de acordo com um critério pré-estabelecido até que uma condição de parada seja atendida.

As árvores de decisão podem ser designadas para tarefas de classificação categóricas e/ou contínuas usando uma função de otimização específica para divisão dos nós, como a função introduzida por Shannon, em 1948, que ficou conhecida por entropia de Shannon: uma fórmula que calcula a incerteza de ocorrência de determinado evento, dada informação parcial sobre o sistema TORRES-GARCíA et al. (2022).

$$E = -\sum_{i=1}^{N} p(x_i) log_2(p(x_i)) \tag{2.2}$$

onde N é o número de classes distintas e p_i é a probabilidade de ocorrência de cada classe. Este valor é maximizado para obter o máximo de informações em cada divisão da árvore de decisão.

Ao construir cada sub-árvore em cada divisão, apenas um conjunto determinado de características são selecionadas e consideradas como candidatas à divisão. Conforme a árvore de

decisão vai se expandindo e criando sub-árvores, mais dificil é extrair a informação interpretável de cada nó. Na figura 2.13, as árvores de decisão recebem a *feature* e definem um cálculo *booleano* para identificar qual nó vai receber a entrada desejada.

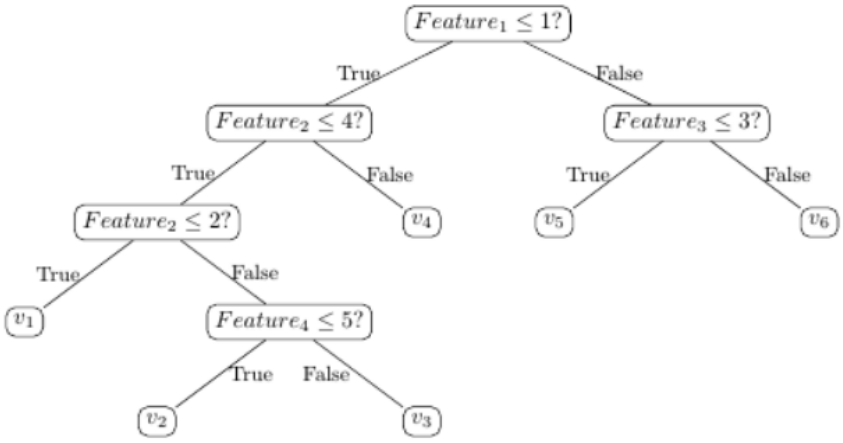


Figura 2.13: Estrutura simplória de uma Random Forest; Imagem extraída de ZHANG (2021).

A árvore de decisão é formalmente expressa com base nesta função:

$$T(x) = \sum_{j=1}^{J} \gamma_j \mathbb{1}(x \in R_j) \tag{2.3}$$

O número de nós-folha J é geralmente tratado como um hiperparâmetro. E a família definida $\{R_j\}_{j=1}^{J}$ é um particionamento do domínio de x. Cada conjunto R_j é parametrizado com θj . A função indicadora $\mathbb{1}(x \in R_j)$ é a função característica do conjunto R_i definido como:

$$\mathbb{1}(x \in R_j) = \begin{cases} 1 & \text{if } (x \in R_j) \\ 0 & \text{otherwise.} \end{cases}$$

Basicamente, o algoritmo *Random forest* soma todas as árvores de decisão com alguns fatores aleatórios. As árvores não apenas extraem algumas amostras do conjunto de treinamento, mas também fazem parte das amostras de *features* durante a indução de cada sub-árvore.

O limite do número de características (parâmetro *max_features*) bem como a quantidade de sub-árvores na árvore de decisão (parâmetro *max_leaf_nodes*) são alguns dos hiper-parâmetros do *Random Forest*. Um valor baixo no parâmetro *max_features* aumenta a chance de seleção de características com baixa relevância, o que consequentemente piora o desempenho da predição nos casos em que seriam mascaradas as características com grandes efeitos, em oposição, um alto valor de *max_features* aumenta o risco de ter apenas candidatos "não-informativos", isto é, com informações não interpretáveis. COURONNÉ et al. (2018)

Uma desvantagem das árvores de decisão é que elas são propensas a superajuste, o que

significa que o modelo se adapta rapidamente ao conjunto de dados, porém, se torna ineficaz na predição de novos resultados. O superajuste das árvores de decisão levará a uma baixa precisão preditiva no geral impactando diretamente na acurácia do algoritmo em classificar corretamente as classes do modelo. SCHONLAU; ZOU (2020)

A Seção 2.4 discutirá o que é sobreajuste e as consequências que pode proporcionar para o classificador ao longo do processo de aprendizagem de máquina.

2.3 Extração de Características

A extração de *features* é uma parte do processo de redução de dimensionalidade, no qual um conjunto inicial de dados brutos é dividido e reduzido a grupos mais gerenciáveis para que diminua a complexidade da fase de processamento. A característica mais importante desses grandes conjuntos de dados é que eles têm um grande número de variáveis. Essas variáveis requerem muitos recursos de computação para serem processadas, portanto, a extração de *features* ajuda a obter o melhor recurso desses conjuntos de *big data*, selecionando e combinando variáveis, reduzindo efetivamente a quantidade de dados. Essas *features* são transcritas em valores numéricos capazes de descrever o conjunto de dados real com precisão e originalidade CHATTERJEE (2022).

Os dados são representados por um número fixo de características que podem ser binário, categórico ou contínuo. Encontrar uma boa representação de dados depende estritamente do domínio e das relações com as medições disponíveis. Por exemplo, em um diagnóstico médico, as características podem ser sintomas, ou seja, um conjunto de variáveis que categorizam o estado de saúde de um paciente (febre, nível de glicose, nível da pressão, etc.).

Há muitas técnicas de extração de *feature*, entretanto, o foco deste trabalho é analisar os procedimentos que usam conceitos matemáticos para extrair os atributos (*features*) do conjunto de dados. Dessarte, as próximas seções ficarão responsáveis de explicar o funcionamento de 3 algoritmos de extração: Transformação Numérica de Fourier, Entropia e Redes Complexas.

2.3.1 Transformação Numérica de Fourier (*Real* e *Z-Curve*)

O teorema da série de Fourier diz respeito à propriedade de sinais periódicos, independente da forma que está expresso o sinal, ele pode ser representado por uma soma de sinusóides, em outros termos, uma série de sinusóides que são iguais ou múltiplos da frequência do sinal. Qualquer sinal periódico pode ser representado de forma equivalente por sinusóides que estão harmonicamente relacionados com a frequência base do sinal. A conversão de um sinal em seu equivalente senoidal é conhecida como **transformação de Fourier** (*FT*). A análise da série de Fourier e a transformada de Fourier não são os únicos caminhos para definir as características de uma frequência ou espectro de um sinal, mas conduzem à um visão mais geral sobre sinais SEMMLOW (2012).

Para colocar o teorema da série de Fourier em termos matemáticos, a forma geral da série é:

$$T(t) = \frac{a_0}{2} + \sum_{n=1}^{\infty} [a_n cos(\frac{nt\pi}{L}) + sin(\frac{nt\pi}{L})] \tag{2.4}$$

em que os coeficiente a_0, a_n, e b_n são números que variam de acordo com a função que será representada, de período fundamental 2L. Esses coeficientes são as amplitudes de cada onda em série, que são calculadas com as seguintes fórmulas:

A maioria das análises de Fourier é aplicada a dados discretos. Os dados discretos diferem dos dados contínuos e periódicos de duas maneiras fundamentais: eles são amostrados por tempo e amplitude e são finito. Os dados finitos podem ser considerados como um sinal aperiódico ou como um período de um sinal periódico.

Para extrair features com base em uma abordagem de Fourier, aplicamos a transformada discreta de Fourier (DFT), amplamente utilizada para imagem digital e processamento de sinal (GSP), que pode revelar periodicidades ocultas após a transformação de dados no domínio do tempo em frequência espaço de domínio. Uma versão discreta das equações de análise de Fourier é a Transformação de Fourier Discreta (DFT) de um sinal, que pode ser definido pela equação:

$$X[k] = \sum_{n=0}^{N-1} x[n] e^{\frac{-j2\pi kn}{N}} \tag{2.5}$$

Onde N é o comprimento do sinal (número harmônico) e a frequência do sinal definida por k. Se os dados são realmente periódicos, esta equação está realizando uma análise discreta da série de Fourier

A transformação de Fourier tem sido amplamente estudado em bioinformática, principalmente para análise de periodicidades e elementos repetitivos em sequências de DNA e estruturas de proteínas YIN; YAU (2005). Para calcular a DFT, usa-se a transformada rápida de Fourier (FFT), que é um método altamente eficiente para calcular a DFT de uma série temporal. No entanto, para usar técnicas *GSP*, uma representação numérica deve ser usada para a transformação ou mapeamento de dados genômicos.

De acordo com MENDIZABAL-RUIZ et al. (2017a), essas representações podem ser divididas em três categorias: mapeamento de valor único, mapeamento de sequência multidimensional e mapeamento de sequência cumulativa. Apesar disso, neste trabalho será estudado apenas duas representações numéricas: **Real** e **Z-curve**.

- Representação Real: Este mapeamento aplica valores decimais negativos para as purinas (A, G) e valores decimais positivos para as pirimidinas (C, T). Por exemplo, digamos que s = (G, A, G, A, G, T, G, A, C, C, A), então, r = = (-0.5, -1.5, -0.5, -1.5, -0.5, 1.5, -0.5, -1.5, 0.5, 0.5, -1.5). A equação deste método é:

$$r[n] = \begin{cases} \text{-0.5}, & \text{s[n] = G} \\ \text{-1.5}, & \text{s[n] = A} \\ \text{0.5}, & \text{s[n] = C} \\ \text{1.5}, & \text{s[n] = T} \end{cases} \quad \text{n = 0, 1, ..., N - 1}$$

$$R[k] = \sum_{n=0}^{N-1} r[n] e^{\frac{-j2\pi kn}{N}}, P_r[k] = |R[k]|^2, k = 0, 1, ..., N-1 \tag{2.6}$$

- Representação Z-curve: O esquema da curva Z é uma curva tridimensional apresentada por MENDIZABAL-RUIZ et al. (2017b) para codificar sequências de DNA com mais semânticas biológicas. Uma dada sequência s[n] de comprimento N, levando em consideração o n-ésimo elemento da sequência (n = 1, 2, ... , N). Denotamos os números de ocorrência cumulativos como A_n, C_n e T_n para cada base nitrogenada A, C, G e T, como o número de ocorrências na sequência, variando de {1, ..., n}. Este método, em sua essência, diminui o número de indicações de sequências de 4 (representação numérica de *Voss*) para 3 (Z-curve) de forma simétrica para todos os 4 componentes, logo:

$$A_n + C_n + G_n + T_n = n \tag{2.7}$$

Onde a curva Z consiste em uma série de nós P1, P2, ... , PN, cujas coordenadas x[n], y[n] e z[n] (n = 1, 2, ... , N) são exclusivamente determinadas pela "transformada-Z", mostrada na equação abaixo:

$$P[n] = \begin{cases} \text{x[n] =} & (A_n + G_n) - (C_n + T_n) \\ \text{y[n] =} & (A_n + C_n) - (G_n + T_n) \\ \text{z[n] =} & (A_n + T_n) - (C_n + G_n) \end{cases} \quad x[n], y[n], z[n] \in [-n, n], \quad \text{n = 1, 2, ..., N}$$

2.3.2 Entropia (*Shannon* e *Tsallis*)

A entropia, como medida de conteúdo e complexidade da informação, foi introduzido pela primeira vez por Shannon (1948), desde então a entropia assumiu muitas formas, nomeadamente topológicas e métricas. Essas entropias foram definidas com o propósito de classificar um sistema através de alguma medida de complexidade ou simplicidade. Essas definições de entropia têm sido aplicadas a sequências de DNA com amplo êxito KOSLICKI (2011).

Uma abordagem algorítmica e matemática para análise de código de DNA usando entropia é descrita como:

$$P_k(s) = \frac{c^k}{N-k+1} = (\frac{c_1^1}{N-1+1}, ..., \frac{c_4^1}{N-1+1}, \frac{c2_{4+1}}{N-2+1}, ..., \frac{c_i^k}{N-k+1}) \quad (2.8)$$

Nesse método, cada sequência é mapeada na frequência das bases vizinhas k, gerando informações estatísticas, e o k-mer é denotado por P_k. A equação é aplicada para cada sequência com frequência de k = 1, 2, ..., 24, onde c_i^k é o número de ocorrências de substrings com tamanho k em uma sequência (s) com tamanho N, o qual o indexador $i \in 1, 2, ..., 4^1 + ... + 4^k$ representa a substring analisada. Como $s \in A, C, G, T$, dado qualquer inteiro positivo k, há 4^k possíveis diferentes valores para *k-mers*. Basicamente, são adotados histogramas com bins curtos, como [A, C, G, T], que ocorrem para k = 1, até histogramas com bins de contagem de sequência, como [GGGGGGGGG, ... ,AAAAAAAAAA], esse resultado para k = 9. Onde, depois de contar o valor absoluto de frequências de cada k, geramos frequências relativas (equação 2.8) e, em seguida, a entropia de Shannon e Tsallis é aplicada para gerar as *features*. BONIDIA et al. (2021a)

A entropia de Shannon é um quantificador estatístico amplamente utilizado para a caracterização de processos complexos. Ele é capaz de detectar aspectos de não linearidade em séries de modelos, contribuindo para uma explicação mais confiável sobre a dinâmica não linear de diferentes pontos de análise, o que, por sua vez, aumenta a compreensão da natureza de sistemas complexos caracterizados por complexidade e não equilíbrio.

Entropia refere-se a uma medida de imprecisão e aleatoriedade em um sistema. Se assumirmos que todos os dados disponíveis pertencem a uma classe, não será difícil prever a classe de um novo dado. A entropia é 0 neste caso. Sendo um valor entre 0 e 1, quando todas as probabilidades são iguais, a entropia assume o seu valor máximo. Dependendo das variáveis, a definição de uma classe com baixa probabilidade é baseada em quão baixa é a probabilidade desse incidente. E isso pode ocorrer quando passar a existir uma classe com pequena probabilidade de realização, o que será aceitável com correlação reversa devido a tal probabilidade. A imprecisão que ocorre quando um evento E ocorre com p probabilidade é denotada por S(p). Se a probabilidade de uma classe ocorrer é 1, a representação é S(1) = 0. De acordo com Shannon, as probabilidades de uma classe ocorrer são $p_1, p_2, p_3, ..., p_n$, a saída examina a imprecisão por meio da medição (H). KARACA; MOONIS (2022)

A entropia de Shannon pode ser denotada como:

$$H(x) = \sum_{i=1}^{n} p_i log_2 p_i \quad (2.9)$$

Aqui, $H(x)$ é definida como a entropia da variável aleatória X. A vagueza média associada a X é interpretada como a vagueza de X.

Além da entropia de Shannon, há também o conceito de entropia de Tsallis, de Constantino Tsallis, que em 1988 reformulou a Entropia de *Boltzmann-Gibs* de forma generalizada com o

uso de uma quantidade normalmente dimensionada em multifractais. As estruturas multifractais têm sua importância em muitas áreas ativas de pesquisa (por exemplo, sistemas dinâmicos não lineares, crescimento de modelos, estruturas comensuráveis/incomensuráveis). Tsallis postulou então a entropia como:

$$S_q = k\frac{1-\sum_{i=1}^{W} p_i^q}{q-1} \tag{2.10}$$

Esta entropia ficou conhecida como **Entropia de Tsallis**. Nela, a quantidade que é normalmente escalada é p_i^q, onde p_i é a probabilidade associada a um evento, $q \in \mathbb{N}$ representa a generalização da equação e $W \in \mathbb{N}$ é o número total de possíveis configurações microscópias.

Em um âmbito voltado a extração de características, ambas têm notoriedade: a entropia de Shannon quantifica a quantidade de informação de uma variável para que consiga chegar a um valor único que mensure a informação contida em diferentes períodos de observação (por exemplo, o *k-mer* mencionado na definição de entropia de Shannon). No entanto, no artigo BONIDIA et al. (2021a), é relevante explorar a forma generalisada da entropia de Shannon para ter mais opções na extração. Deste modo, para uma variável aleatória discreta F tomando valores em $\{f[0], f[1], f[2], ..., f[N-1]\}$ com probabilidades $\{p[0], p[1], p[2], ..., p[N-1]\}$, representado por $P(F = f[n]) = p[n]$. A entropia de Shannon e Tsallis associada com essa variável é dada pelas expressões a seguir:

$$H_s[k] = -\sum_{n=0}^{N-1} p_k[n] log_2 p_k[n]; k = 1, 2, ..., 24 \tag{2.11}$$

$$H_t[k] = \frac{1}{q-1}(1 - \sum_{n=0}^{N-1} p_k[n]^q); k = 1, 2, ..., 24 \tag{2.12}$$

2.3.3 Redes Complexas

O estudo de redes complexas é inspirado em estudos de análise empírica em redes reais. De fato, redes complexas permitem compreender vários sistemas reais, desde redes tecnológicas para redes biológicas. Por exemplo, nós precisamos de um conjunto de neurônios conectados por sinapses para garantir nossa capacidade de ler este texto; nosso corpo é governado por interações entre milhares de células; infraestruturas de comunicação como a Internet são formadas por roteadores e cabos de fibras ópticas e a sociedade é composta por pessoas conectadas por relações sociais, colaborações entre familiares e/ou profissionais CALDARELLI (2007).

Para CALDARELLI (2007), esses sistemas são chamados de sistemas complexos porque não é possível prever seu comportamento coletivo a partir de componentes individuais, mas entender a correlação matemática desses sistemas nos torna capazes de prevê-los e possivelmente controlá-los. Em geral, um modelo de rede produz grafos com propriedades semelhante ao sistema real. No entanto, a vantagem de usar um modelo é reduzir a complexidade do mundo real a um nível que pode ser tratado de forma mais prática. Portanto, as redes são consideradas

um poderoso meio de representação padrões de conexões entre partes de sistemas, como Internet, rede elétrica, teias alimentares e redes sociais.

Do ponto de vista matemático, podemos representar uma rede por meio de uma matriz de adjacência **A**. Um gráfico de N vértices tem uma matriz de adjacência $N \times N$. As arestas podem ser representada pelos elementos A_{ij} desta matriz tal que:

$$A_{ij} = \begin{cases} 1, & \text{se os vértices i e j estão conectados} \\ 0, & \text{caso contrário} \end{cases}$$

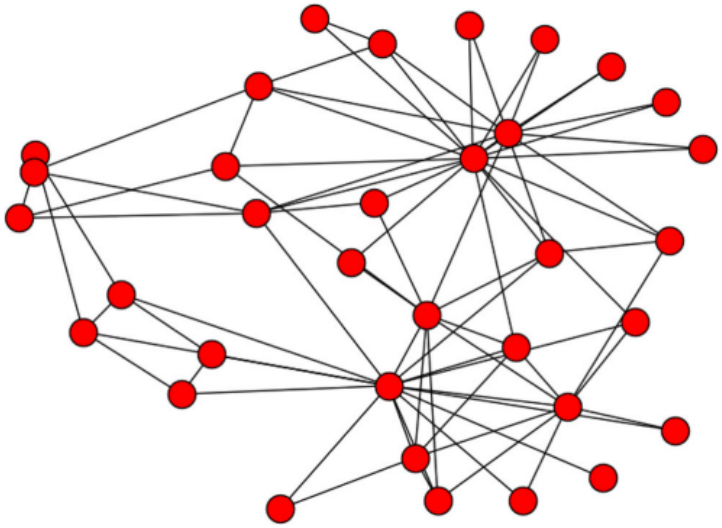


Figura 2.14: Uma representação em rede das interações sociais de um grupo que pertence ao mesmo clube de karatê; De acordo com MATA (2020), essa rede social foi estudada por ZACHARY (1977) de 1971 a 1972 e capta os links de 34 sócios que se integraram entre si fora do clube

Geralmente, as matrizes adjacentes no mundo real são assimétricas, ou seja, $A_{ij} \neq A_{ji}$, porém na imagem 2.15 o grafo é simétrico. Se considerarmos redes com peso, então, cada aresta terá um diferente peso w_{ij}. O peso pode representar o fluxo de pessoas em um voo em um transporte ou a corrente que flui através de uma linha de transmissão em uma rede elétrica. Nestes casos, os elementos da matriz de adjacência são melhor descritos como $A_{ij} = w_{ij}$, e, geralmente, $0 \leq w_{ij} \leq 1$.

Uma informação relevante que pode ser obtida na matriz de adjacência é o grau k_i de um vértice i definido como o número de arestas ligadas ao vértice i, ou seja, o número de vizinhos mais próximos do vértice i. O grau dos vértices pode ser escrito por meio da matriz de adjacência como:

$$k_i = \sum_{j=1}^{N} A_{ij} \tag{2.13}$$

Uma questão central na estrutura de um grafo é a conectividade de seus vértices, ou

seja, a possibilidade de estabelecer um caminho entre quaisquer dois nós. Isso é importante, por exemplo, quando temos um impulso nervoso se propagando em um rede neural. CALDARELLI (2007)

Normalmente, o número de vizinhos em uma rede complexa a uma distância pode ser aproximado por $\langle k \rangle^l$, considerando que cada vértice tem grau igual ao grau médio de a rede $\langle k \rangle$ e não há *loop*. Um *loop* ou um ciclo é um caminho fechado P_{ij} (i = j) no qual todos os nós e todas as arestas são distintas.

Em meio a tantas propriedades de grafos, basta entender estas que foram discutidas para entender de maneira simplória o funcionamento de uma rede complexa na área de ML no que diz respeito à extração de *features*.

No contexto de extração de características, as sequências são mapeadas para a frequência de vizinhos com base nitrogenada k. Este mapeamento é convertido em um grafo não direcionado representado por uma matriz de adjacência, na qual é aplicado um esquema de *threshold* (limite) para extração de *features*, gerando assim um vetor de *features* adaptado. Um grafo $G = \{V, E\}$ é estruturado por um conjunto V de vértices (ou nós) conectados por um conjunto E de arestas (ou *links*) e cada aresta é conectada a dois vértices.

Nesta circunstância, o grafo é não direcionado, o que implica que a matriz de adjacência A é simétrica, então os elementos $a_{ij} = a_{ji}$ para qualquer i e j. É aplicado então o esquema de *threshold* (limite) e para cada *threshold* (t), um novo subgrafo é gerado do grafo original. Este procedimento se propõe a capturar as adjacências em diferentes frequências para que várias medidas de caracterização da rede (grafo e subgrafos), entre eles: intermediação, assortatividade, grau médio, comprimento médio do caminho, grau mínimo, grau máximo, desvio padrão de grau, frequência de *motifs* e coeficiente de agrupamento sejam capturadas. Todos esses fatores serão usados como os atributos do conjunto (*features*). BONIDIA et al. (2021a)

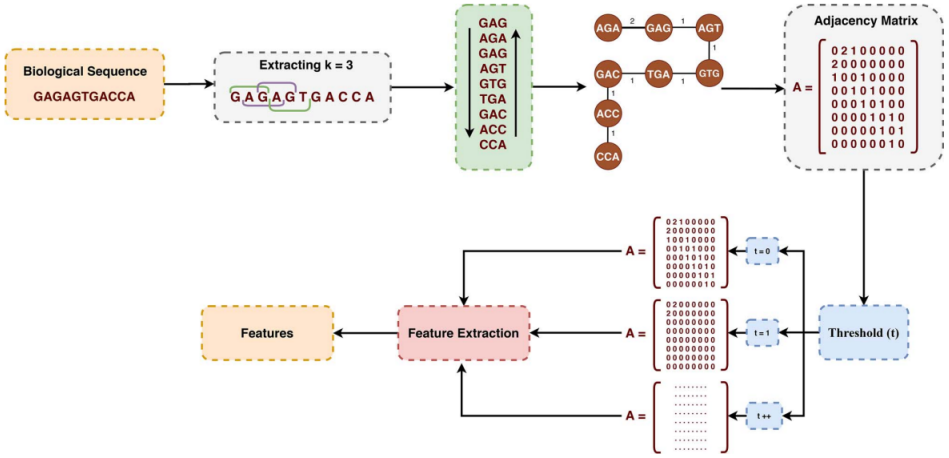


Figura 2.15: Fluxo de uma rede complexa. (1) Cada sequência é mapeada em cada frequencia de vizinhos de base k = 3; (2) O mapeamento é convertido em um grafo não direcionado representado por uma matriz de adjacência; (3) A extração de *features* é executada usando o esquema de *threshold*; (4) As *features* são geradas. A imagem foi extraída de BONIDIA et al. (2021a).

2.4 *Overfitting* (Sobre-ajuste)

Por ser um problema bem comum na área de *Machine Learning*, o sobreajuste de dados ocorre quando o modelo se adapta extremamente bem ao conjunto de treinamento, generalizando o modelo a partir dos dados observados e não-observados, porém se adequa mal ao conjunto de testes. Modelos superajustados tendem a memorizar a entrada baseado em um *bias*, incluindo ruído inevitável no conjunto de treinamento, em vez de aprender as características relevantes específicas de cada entrada.

O *overfitting* geralmente ocorre quando o modelo é muito complexo e pode ser identificado a partir destas duas características:

- O modelo tenta memorizar os dados de treinamento em vez de aprender padrões essenciais dos dados.
- O modelo tem um bom desempenho apenas nos dados de treinamento e um desempenho ruim em novos dados não vistos.

De acordo com PRAMODITHA (2022), uma maneira eficaz de reconhecer se um modelo está superajustado é traçar a **curva de aprendizado** que irá apontar o *overfitting* nos modelos de aprendizado profundo. A curva de aprendizado é um gráfico bidimensional que descreve em porcentagem a taxa de melhoria de um certo modelo. Em outras palavras, ela pontua o modelo de treino e a sua validação em relação à um número de lotes ou picos, chamado de *epoch*.

A curva indica que o modelo está superajustado caso satisfaça duas condições:

- Se houver uma lacuna clara entre as pontuações de treinamento e validação.
- Quando o erro de validação (*loss*) começa a aumentar em algum ponto enquanto o erro de treinamento (*perda*) ainda diminui. No caso da precisão, a precisão da validação começa a diminuir em algum ponto enquanto a precisão do treinamento ainda aumenta.

Como ilustração, a figura 2.16 menciona essas duas condições.

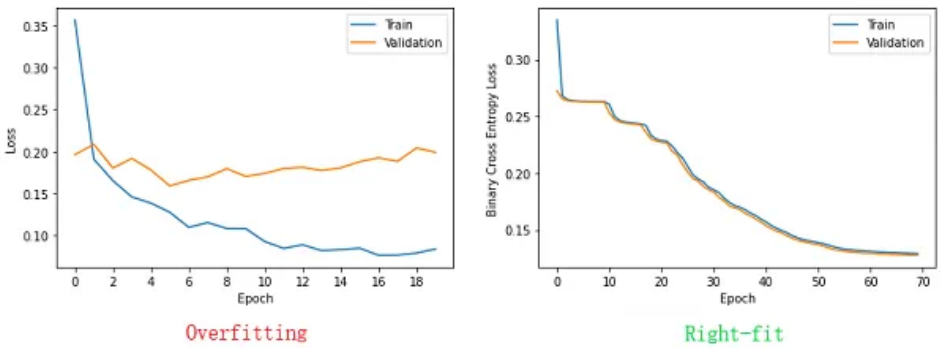


Figura 2.16: Curva de aprendizado; Imagem extraída de PRAMODITHA (2022).

Para detectar *overfitting* em métodos de aprendizado de máquina gerais como árvores de decisão, Random Forest e até mesmo *K-Nearest Neightbors*, há outro gráfico chamado de **curva de validação**.

A curva de validação representa a influência de um único hiperparâmetro no conjunto de treinamento e validação. Para isso, é necessário identificar o hiperparâmetro mais importante do modelo e traçar a influência de seus valores usando a curva de validação. Por exemplo, na figura 2.17, o algoritmo *Random Forest* com o hiperparâmetro *max_depth*, que representa a profundidade da árvore de decisão, pode ser usado como parâmetro de verificação de influência:

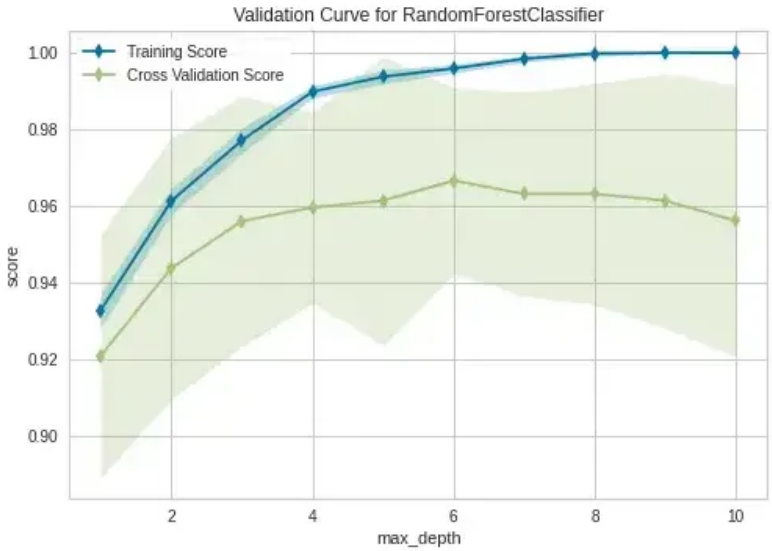


Figura 2.17: Curva de validação; Imagem extraída de PRAMODITHA (2022).

O eixo x representa os valores do hiperparâmetro fornecido, enquanto o eixo y representa as pontuações de treinamento e validação em porcentagem. Neste gráfico em questão, após o *max_depth* ser maior que 6, o modelo começa a sobreajustar os dados de treinamento. Em outras palavras, a precisão da validação começa a diminuir em *max_depth=6* enquanto a precisão do treinamento ainda aumenta.

Há também como verificar a acurácia da fase de treinamento e da validação pela abordagem da matriz de confusão, observando os valores de cada quadrante. A ideia é que se o conjunto de treinamento tiver uma acurácia muito alta em comparação com o conjunto de testes, possivelmente é um indicador que o modelo não se performou bem ao predizer dados não-vistos e portanto, ocorreu um sobreajuste.

Para lidar com o *overfitting*, PRAMODITHA (2022) cita alguns métodos importantes que previnem e atacam diretamente nesta problemática:

1. **Redução de dimensão**: A redução do número de *features* nos dados é chamada de redução de dimensionalidade. Devemos manter o máximo possível de variação nos dados originais. Caso contrário, perdemos informações úteis nos dados.

2. **Seleção de features**: A seleção de *features* pode ser considerado como um método de redução de dimensionalidade, pois remove recursos redundantes (desnecessários) do conjunto de dados. Isso reduz o número de *features* e, consequentemente, reduz a dimensão dos dados.

3. **Parada antecipada**: Na parada antecipada, o processo de treinamento é parado intencionalmente antes que o modelo comece a se sobreajustar, observando a curva de aprendizado ou a curva de validação.

4. **Validação Cruzada *k-fold***: Na validação cruzada *k-fold*, o conjunto de dados completo é dividido em diferentes partes iguais, dependendo do valor de k (geralmente 5 ou 10). Cada parte contém diferentes tipos de instâncias (pontos de dados) e para cada pedaço é feito uma avaliação de métrica.

5. **Limitação de conjunto**: Um *ensemble* (grupo) é uma coleção de várias árvores de decisão criadas a partir de subconjuntos dos dados e *features* de treinamento. Por exemplo, uma *Random Forest* é um *ensemble* que contém um grupo de árvores de decisão não correlacionadas, e devido a essa característica a aleatoriedade extra ocorre. Além disso, o resultado final é calculado pela média dos resultados de cada árvore não correlacionada. Assim, a *Random Forest* produz resultados mais precisos e estáveis do que uma única árvore de decisão.

6. ***Pre-prunning***: Por padrão, uma árvore de decisão é desenvolvida até o máximo de sua profundidade, e caso seja atingido o seu limite, ocorrerá o *overfitting* dos seus dados. Nas árvores de decisão, o *pré-prunning* é o processo de controle do crescimento da árvore. O *pré-prunning* aplica uma regra de parada antecipada que interrompe o crescimento de uma árvore de decisão previamente, ficando com menos ramificações do que o esperado. Para que seja aplicado esta técnica, os hiperparâmetros *max_depth*, *min_samples_leaf*, *min_samples_split* são limitados.

7. ***Pós-pruninng***: O *pós-prunning* é o processo de remoção de partes da árvore após a árvore ter crescido completamente.*Cost complexity pruning* (ccp_alpha) é um hiperparâmetro deste método que aumenta o número de nós reduzindo a profundidade da árvore. Logo, valores maiores de *ccp_alpha* evitam o *overfitting*.

8. **Regularização de ruído**: Ao adicionar ruído aos dados de treinamento, uma pequena quantidade de ruído será adicionada a cada instância de treinamento e gerará diferentes versões da mesma instância, expandindo cada conjunto. Essa camada extra é comumente usada em algoritmos de rede neural pois ajuda a evitar o *overfitting*.

9. **Regularização de "abandono"**: É um método de regularização específico da rede neural que remove aleatoriamente alguns nós da rede durante o treinamento com base

no valor da probabilidade que definimos em cada camada. Os nós removidos não participam do processo de atualização dos parâmetros e a regularização é aplicada por camada. A rede original torna-se menor após a aplicação deste algoritmo e, assim são menos flexíveis.

No artigo de DEMŠAR; ZUPAN (2021), os autores enfatizam que a modelagem não se limita a ajustar os parâmetros do modelo, mas inclui todos os outros procedimentos como pré-processamento de dados, seleção de modelo e ajuste de hiperparâmetros. O experimento feito demonstrou que é inevitável não ter esta etapa de pré-processamento de dados antes de alimentar os dados em um algoritmo de aprendizagem, pois o resultado final do algoritmo depende estritamente desta metodologia para ser útil.

2.5 *Cross-Validation* (Validação Cruzada)

A Validação Cruzada (CV) é uma técnica de reamostragem usada para avaliar modelos de ML em uma amostra limitada de dados ou desconhecido dados que ajudariam a fazer previsões sobre dados que não foram usados durante o ciclo de treinamento. Segundo RABELLO (2019), o CV consiste em particionar os dados em conjuntos (partes), onde um conjunto é utilizado para treino e outro conjunto é utilizado para teste e avaliação do desempenho do modelo. Este procedimento tem altas chances de detectar se o seu modelo está sobreajustado aos seus dados de treinamento, ou seja, provocando *overfitting*. O uso de subconjuntos aleatórios de dados em validação cruzada, também conhecida como validação cruzada k-fold, é uma forte maneira de testar a taxa de sucesso dos modelos usados para classificação MüLLER (2020).

Na validação cruzada, um conjunto de dados D é particionado em ***k* dobras** (subconjuntos disjuntos) D_i para i = {1, ..., k}. A cada iteração, treino e teste, um conjunto formado por K-1 subconjuntos são utilizados para treinamento e o subconjunto restante será utilizado para teste gerando um resultado de métrica para avaliação (ex: acurácia). Um particionamento aleatório clássico do conjunto de dados em k dobras permite que haja registros presentes simultaneamente tanto no treinamento quanto na validação, induzindo assim o vazamento de dados e gerando estimativas superotimistas de desempenho em comparação com a validação cruzada sem duplicatas, que dispõe a estimativa imparcial. GUO (2021)

O processo *k-fold* é um mecanismo que minimiza as desvantagens do método *hold-out*, que é uma metodologia da qual divide o conjunto de dados em duas partes: o conjunto de treinamento e o conjunto de teste e normalmente, 80% do conjunto de dados vai para o conjunto de treinamento e 20% para o conjunto de teste nos diversos casos, todavia, são parâmetros ajustáveis e fica a cargo do programador adaptá-los. Apesar da facilidade do método *hold-out*, se o conjunto de dados não for completamente uniforme ou bem distribuído, pode ser que ao separar os dois conjuntos (de treinamento e de testes) ocorra uma divergência significativa entre os conjuntos, o que reflete na precisão da acurácia do modelo.

A figura 2.18 apresenta uma nova maneira de dividir o conjunto de dados que supera as consequências do método clássico *hold-out* que baseado no referencial LYASHENKO (2023) funciona da seguinte forma:

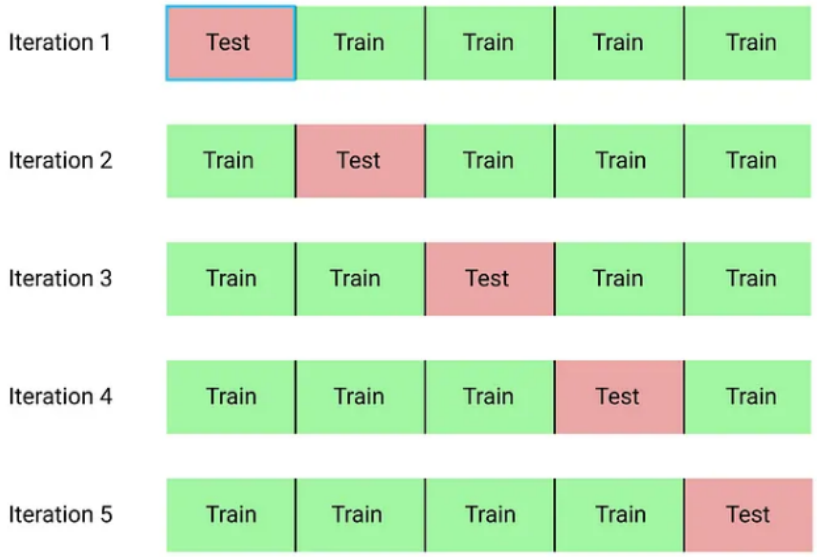


Figura 2.18: Procedimento *k-fold*; Imagem extraída de SHAIKH (2018).

1. Escolha um número de *folds-k*, neste caso, k = 5.
2. Divida o conjunto de dados em k partes iguais chamadas de dobras, se possível.
3. Escolha k – 1 dobras como o conjunto de treinamento. A dobra restante será o conjunto de teste.
4. Treine o modelo no conjunto de treinamento. A cada iteração de validação cruzada, você deve treinar um novo modelo independentemente do modelo treinado na iteração anterior.
5. Valide o conjunto de teste.
6. Salve o resultado da validação.
7. Repita os passos {3, ..., 6} ***k* vezes**. A cada vez, use a dobra restante como conjunto de teste. No final, o modelo deve ter sido validado em todas as dobras que possui.
8. Para obter a pontuação final, a média dos resultados obtidos na etapa 6 deve ser calculada.

O *K-Fold* estratificado é uma variação da técnica padrão de *k-Fold* CV, projetada para ser eficaz nesses casos de desequilíbrio do valor alvo. A técnica, comumente chamada de Validação Cruzada Estratificada (SKCV), garante que as frequências de classe relativas sejam efetivamente

sustentadas em cada divisão na fase de validação quando usadas amostragens estratificadas em vez de amostragens aleatórias, principalmente em problemas de classificação PRUSTY; PATNAIK; DASH (2022).

Este método, que usa amostragem estratificada, divide o conjunto de dados **k dobras** de modo que cada parte contenha aproximadamente a mesma porcentagem de amostras de cada classe de destino que o conjunto completo. No caso de regressão, o SKCV garante que o valor alvo médio seja aproximadamente igual em todas as dobras.

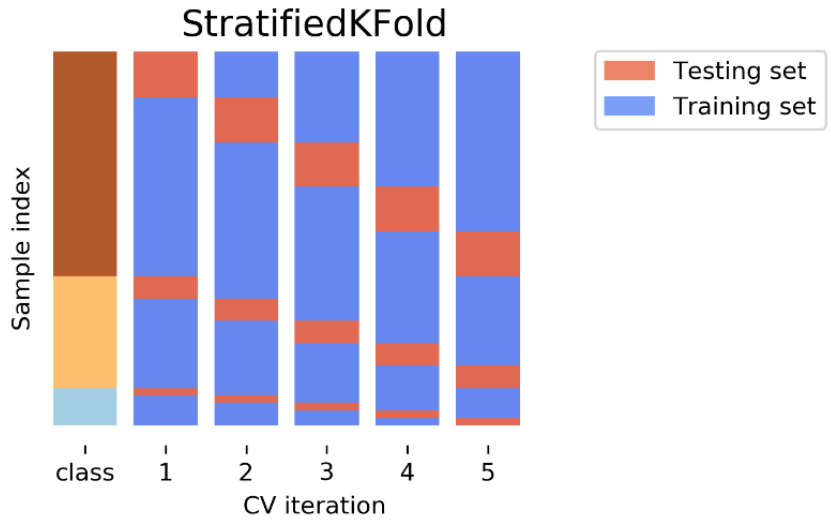


Figura 2.19: Método SKCV; Imagem extraída de SHAIKH (2018). O eixo x representa as iterações do CV e o eixo y o indexador dos conjuntos.

O SKCV preserva as frequências de classe em cada dobra para serem as mesmas do conjunto de dados geral. A figura 2.19 é um exemplo de um conjunto de dados com três classes ordenadas. Se for aplicado um k-3 ao algoritmo, o primeiro terço dos dados estaria na primeira dobra, o segundo na segunda dobra e assim por diante. Como esses dados estão ordenados, a validação cruzada comum seria ruim pois irá perder a consistência entre os dados, ao contrário do SKCV que garante que cada parte terá exatamente 1/3 dos dados de cada classe.

2.6 Métricas de avaliação

Avaliar o algoritmo de aprendizado de máquina é uma parte essencial de qualquer projeto de ML, pois para verificar se o classificador conseguiu predizer corretamente o conjunto de treinamento, é preciso mensurar a sua competência de predizer o modelo. A precisão da classificação é usada para medir o desempenho do modelo, todavia, nem sempre a avaliação será satisfatória e os resultados vão variar de acordo com a métrica escolhida. Por exemplo, a métrica *precisão_score* pode, em alguns casos, fornecer resultados ruins quando avaliado em relação a outras métricas.

Uma métrica exclusiva não é o suficiente para determinar a acurácia correta do modelo, diante disso, este tópico retratará as diferentes métricas existentes para calcular a eficiência do

modelo preditivo, levando em conta como é feito a estimativa e a sua relação com a matriz de confusão.

A matriz de confusão exibirá a distribuição dos registros em termos de suas classes atuais e de suas classes previstas e detalha em como o classificador preveu o conjunto. A tabela 2.1 mostra as frequências de classificação para cada classe do modelo por meio dos valores de *TP, TN, FP, FN*.

Em que TP, FP, TN e FN significa respectivamente:

- Verdadeiro positivo (*true positive* — TP): quando o método diz que a classe é positiva e, ao verificar a resposta, vê-se que a classe era realmente positiva.
- Falso positivo (*false positive* — FP): quando o método diz que a classe é positiva, mas ao verificar a resposta, vê-se que a classe era negativa;
- Verdadeiro negativo (*true negative* — TN): quando o método diz que a classe é negativa e, ao verificar a resposta, vê-se que a classe era realmente negativa;
- Falso negativo (*false negative* — FN): quando o método diz que a classe é negativa, mas ao verificar a resposta, vê-se que a classe era positiva;

Tabela 2.1: Representação em tabela da matriz de confusão com seus respectivos valores booleanos.

Conjunto	TN	FP	FN	TP
Classe real	0	0	1	1
Classe prevista	0	1	0	1

Ao término de cada cálculo de métrica, o algoritmo de treinamento irá efetuar a avaliação do melhor modelo preditivo baseado na melhor taxa de *score*.

Há métricas como o *Precision*, *Recall*, *F1*, *F-Beta* e o *Roc-AUC*. E, no geral, a matriz de confusão é a base de cálculo para estas métricas.

O *precision-score* é uma métrica de classificação que mede a capacidade de um classificador de não rotular como positiva uma amostra negativa. Ou seja, se o classificador tiver muitos acertos sobre o conjunto verdadeiro-positivo, isso resultará em uma pontuação de precisão mais alta. Quanto maior o valor da métrica, melhor. O melhor valor possível é 1 (se um modelo acertou todas as previsões) e o pior é 0 (se um modelo não fez uma única previsão correta).

$$Precision = \frac{TP}{TP + FP} \tag{2.14}$$

O *recall-score* é uma métrica de classificação que apresenta uma proporção de previsões da classe verdadeira-positiva em relação ao número total de amostras positivas. Em outras

palavras, *recall* mede a capacidade de um classificador para detectar amostras positivas sobre um modelo.

$$Recall = \frac{TP}{TP+FN} \tag{2.15}$$

A pontuação F1 pode ser interpretada como uma média harmônica do *Precision* e *Recall* pois a contribuição relativa de ambas é igual. A equação do *F1-score* é:

$$F1 = 2 \cdot \frac{(precision \cdot recall)}{(precision + recall)} \tag{2.16}$$

A pontuação F-beta é a média harmônica ponderada de *precision* e *recall*, atingindo seu valor ideal em 1 e seu pior valor em 0.

$$F_b = \frac{1+\beta^2 \cdot (precision \cdot recall)}{(\beta^2 \cdot precision) + recall} \tag{2.17}$$

O parâmetro **beta** representa a razão entre o *recall* e o *precision*. **beta** > 1 dá mais peso ao *recall*, enquanto **beta** < 1 favorece o *precision*. Assintoticamente, se o beta tender à *+inf* considera-se apenas o *recall* e beta tendendo 0 apenas *precision*

A curva *ROC* ou "Curva Característica de Operação do Receptor" é um gráfico que permite avaliar um classificador binário levando em consideração a taxa de verdadeiros positivos (TP) e a taxa de falsos positivos (TF). Essas taxas também podem ser referidas pelas siglas TPR (True Positive Rate) e FPR (False Positive Rate), respectivamente. Esse gráfico permite comparar diferentes classificadores e definir qual o melhor com base em diferentes pontos de corte. Na prática, quanto mais próximo do topo do eixo Y melhor o classificador DöRING (2018). O ROC possui dois parâmetros:

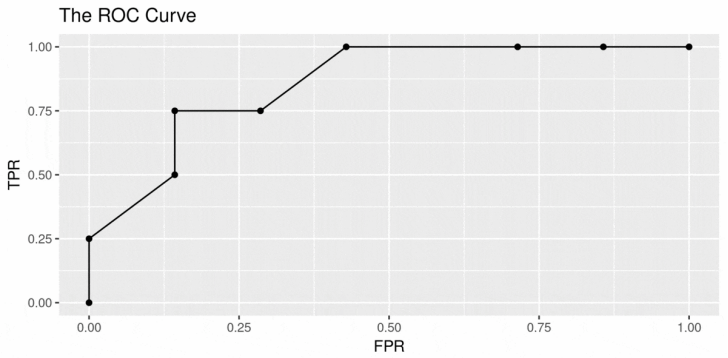
- **Taxa de verdadeiro positivo** (do inglês, *True Positive Rate*), que é dado por:

$$TPR = \frac{TP}{TP+FN} \tag{2.18}$$

- **A Taxa de falso positivo** (*do inglês, False Positive Rate*), que é definida da seguinte maneira:

$$FPR = \frac{FP}{FP+TN} \tag{2.19}$$

Uma curva ROC é um plano bidimensional em relação as duas taxas (TPR e FPR) nos diferentes limiares de classificação. Reduzir o limite de classificação determina mais itens como positivos, o que aumenta os falsos positivos e verdadeiros positivos. Uma curva típica de ROC é apresentada como:



Estimate	-3.0	-2.5	-1.0	-1.0	-0.5	-0.5	0.5	1.0	2.0	2.0	3.5
Class	-1	-1	-1	-1	-1	+1	-1	+1	-1	+1	+1
Prediction	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1
Status	FP	FP	FP	FP	FP	TP	FP	TP	FP	TP	TP

Figura 2.20: Curva ROC e as previsões da classe positiva (TPs) e falsos positivos (FPs)
GOLDSTEIN-GREENWOOD (2022)

AUC significa "área"sob a curva ROC, ela é a derivada da curva ROC. Nesta métrica, há a soma da região abaixo da curva ROC inteira, como se fosse o cálculo de uma integral definida. O valor do AUC varia de 0,0 até 1,0 e o limiar entre a classe é 0,5. Portanto, acima desse limite, o algoritmo classifica em uma classe e abaixo em outra classe.

Uma maneira de interpretar a AUC é a probabilidade de o modelo classificar um exemplo positivo aleatório mais alto do que um exemplo negativo aleatório. Por exemplo, considerando em ordem crescente de previsões de regressão logística:

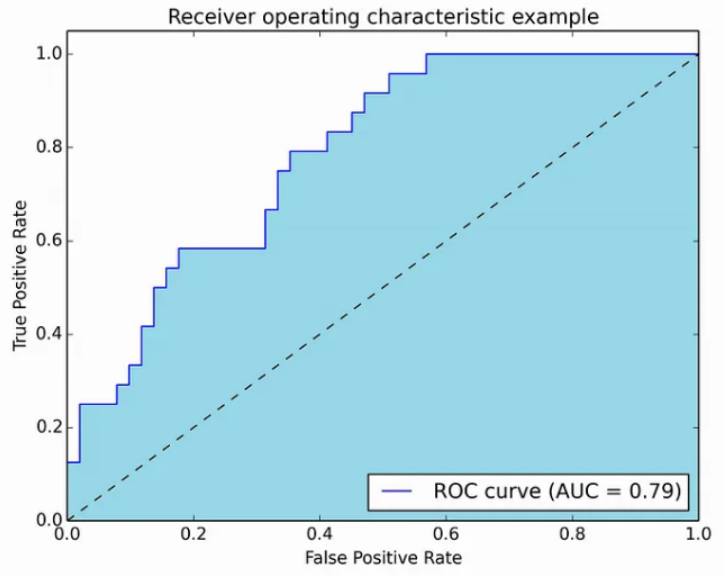


Figura 2.21: Curva AUC abaixo da curva ROC. Extraída de DERNONCOURT (2015)

A AUC pode ser aplicável conforme o objetivo a priori, sendo recomendável por dois motivos:

- É invariante em escala, uma vez que trabalha com precisão das classificações ao invés de seus valores absolutos.
- Mede a qualidade das previsões do modelo, independentemente do limiar de classificação

Apesar de útil na invariância dos resultados e na independência do limiar para a medição da qualidade de previsão do modelo, há ressalvas que podem limitar o uso do AUC em ocasiões específicas:

1. Nem sempre uma variação é adequada quando se trata de escalonamentos: O objetivo do classificador é equilibrar as probabilidades de cada TPRs e FPRs a cada iteração.
2. A incompatibilidade de limite de classificação nem sempre é desejada: Nos casos em que há amplas disparidades no custo de falsos negativos em comparação a falsos positivos, pode ser essencial minimizar um tipo de erro de classificação.

3

Revisão da literatura

A construção do estado de conhecimento teve como princípio a análise sistemática de dissertações, teses, trabalhos científicos e artigos produzidos em um lapso temporal de 6 anos. A estratégia utilizada para a revisão sistemática da literatura busca seguir os critérios adotados por KITCHENHAM; BRERETON (2013) tendo o trabalho de BONIDIA et al. (2021a) como exemplo.

3.1 Questões de pesquisa

As questões de pesquisa norteam a revisão sistemática e têm como objetivo definir a parametrização da problemática identificando os trabalhos que propunham a extração de características em sequências de RNAs, quais métodos de extração matemáticos bem como o comparativo da técnica biológica em detrimento dos modelos matemáticos e a acurácia de cada método considerando um grupo de RNAs não-codificantes (ncRNAs). Portanto, as Questoes de pesquisa (QP) foram definidas a seguir:

- QP_1 Quais os métodos de extração de características em sequências de RNAs?
- QP_2 Quais os modelos matemáticos utilizados na extração?
- QP_3 Quais os grupos de ncRNAs a serem trabalhados e os modelos matemáticos aplicados?

3.2 Estratégia de busca

As bases de dados *PubMed Central*, repositório da UnB, *Oxford Academic*, *Medline*, SIABI/IFB foram consumidas para embasamento teórico e argumentativo da tese. O *PubMed Central* é um banco de dados digital gratuito de literatura científica na área de biomedicina e tecnologia gerenciado e desenvolvido pela *National Library of Medicine* que contém um vasto repositório de artigos científicos mundialmente reconhecido. O repositório da UnB é um serviço digital oferecido pela Biblioteca Central para a gestão e disseminação da produção científica da Universidade de Brasília. A base da *Oxford Academic* publica os periódicos de

cunho científico geral para o público em mais alta qualidade, dispondo de uma comunidade acadêmica da Universidade de Oxford. A Medline que é uma biblioteca virtual de medicina a qual detém os dados indexados por uma palavra-chave específica do sistema *MeSH* e, por fim, o SIABI/IFB, biblioteca virtual do IFB que disponibiliza os recursos dos campus existentes em Brasília.

Tabela 3.1: Base de dados consumidas

Base de dados	**Link para acesso**
PubMed Central	<https://pubmed.ncbi.nlm.nih.gov/>
Repositório UnB	<https://repositorio.unb.br/>
Oxford Academic	<https://academic.oup.com/journals>
Medline	<http://bases.bireme.br/>
SIABI/IFB	<http://siabi.ifb.edu.br/>

Para cada base de dados escolhida foram realizadas buscas avançadas em suas ferramentas de pesquisa com um intervalo de tempo de 6 anos até a data de realização desta revisão (24 de junho de 2022), contemplando como palavras-chaves de pesquisa: *ncRNAs*, *machine learning*, *feature extraction*, *sequence features*, *mathematical approach* às quais resultaram em um conjunto de mais de 300 literaturas. Visando diminuir o escopo das produções para a problemática em questão, modificou-se o critério de análise que apenas considerava o título e resumo dos materiais e passou-se a levar em conta apenas os trabalhos que continham ncRNAs como objeto de estudo. Na seção posterior, mais especificamente no processo de seleção e exclusão, ao passar pela crítica qualitativa das obras, as literaturas serão menos abrangentes e mais voltadas ao estudo de caso da tese. A tabela 3.2 mostra a quantidade de artigos científicos retornados por cada banco de dados no campo de busca.

Tabela 3.2: Resultado das buscas nos bancos de dados

Base de dados	Palavras-chaves	Produções científicas
PubMed Central	*Machine learning, sequence features, ncRNAs*	98
Repositório UnB	*Machine learning, ncRNAs*	5
Oxford Academic	*Machine learning, ncRNAs, mathematic sequence features*	153
Medline	*Machine learning, ncRNAs, mathematic sequence*	34
SIABI/IFB	*Machine learning*	2
Total		292

3.3 Critério de inclusão e exclusão

Para responder as QPs definimos Critérios de Inclusão (CIs) e Critérios de Exclusão (CEs) que irão filtrar os resultados das pesquisas. Os CIs estão listados a seguir.

- Critério de inclusão $(CI)_1$ Produções científicas que usam os ncRNAs como objeto de pesquisa para a extração de características;
- CI_2 Estudos primários que aplicam modelos preditivos supervisionados ou não supervisionados sendo biológico, híbrido ou matemático para classificação de ncRNAs;
- CI_3 Estudos que classificam as classes e grupos de ncRNAs aplicando o modelo matemático de extração de características;

Os Critério de exclusão (CE) irão ajudar a filtrar apenas os artigos científicos relevantes para a revisão. Baseado nas questões de pesquisa que norteam o trabalho, os CEs propostos abaixo selecionarão um grupo concreto de produções a fim de diminuir a abragência e a generalização do tema.

- Estudos que não estejam escritos em português ou inglês;
- Estudos que a versão completa não é disponível gratuitamente;
- Estudos "duplicados", que foram obtidos através da busca em mais de uma base, nestes casos apenas o primeiro será considerado.
- Produções científicas que não classificam o grupo de ncRNAs;
- Estudos descritivos de funcionalidades que não discorre sobre a metodologia de Aprendizado de máquina (ML) empregue;

3.4 Análise e discussão das literatuas

As aplicações do AM extraem informações relevantes de sequências baseadas em várias propriedades biológicas e físico-químicas, usando quadros de leitura abertos (ORF), frequência de uso de nucleotídeos adjacentes, conteúdo *GC* e entre outros. Essas abordagens são comuns em problemas biológicos, mas essas implementações são muitas vezes difíceis de reutilizar ou adaptar a outro problema específico. Um grande exemplo é que os recursos ORF são um guia essencial para ncRNAs de genes codificadores de proteínas, mas não são capazes de classificar classes para os ncRNAs, e, como dito por SZCZEÅNIAK et al. (2020), consequentemente, a extração de um conjunto de características que contêm informação discriminatória significativa para identificá-las é prejudicada, o que influencia na construção de um modelo preditivo.

BONIDIA et al. (2021a) propõe um modelo preditivo matemático para identificação de classes de ncRNAs. Este trabalho foi dividido em três estudos de caso: (I) Avaliação das abordagens matemáticas com os problemas mais frequentes da classe de ncRNAs, por exemplo, *lncRNA versus mRNA*; (II) Teste de generalização em diferentes classificadores de ncRNAs; (III) Análise de persistência em cenários com dados desbalanceados. As técnicas de ML aplicadas consistem na transformação discreta de Fourier, mapeamento numérico (representação de Voss, de Real, de z-curve, de EIIP e de números complexos), entropia de Shannon e Tsallis e o uso de redes complexas.

WANG et al. (2014) em contra-proposta aplica um Algoritmo genético (GA) junto a uma SVM que implementa o método de aprendizado de máquina supervisionado baseado no conceito da teoria de Darwin, isto é, o conjunto de sequências que mais se adaptam a parametrização de classificação dos algoritmos são herdadas na próxima geração a partir do mecanismo de competição. Em suma, a classificação executa um modelo preditivo biológico na categorização do grupo de ncRNAs.

BONIDIA et al. (2021b) apresenta um pacote de 20 descritores matemáticos divididos em 5 grupos: mapeamento numérico, *chaos game*, transformação de Fourier, entropia e grafos. Similar ao seu estudo comparativo BONIDIA et al. (2021a), o autor executa o estudo de caso nos ncRNAs treinando o algoritmo *CatBoost* para classificação de classes e concluiu que a abordagem matemática trouxe uma eficácia significativa nos resultados.

ARAUJO (2016) busca classificar as classes de snoRNAs *(H/ACA box snoRNA e C/D box snoRNA)* empregando uma técnica mais sofisticada na fase de treinamento no intuito de encontrar bons meta-parâmetros da SVM. A ideia é usar o Explicit Decomposition with Neighborhoods (EDeN), um kernel decomposicional de grafos baseado no Neighborhood Subgraph Pairwise Distance Kernel (NSPDK), que pode ser usado para a geração explicita de features a partir de grafos e as SVMs que geram um hiperplano como superfície de decisão de tal modo que a margem de separação entre amostras positivas e negativas é maximizada formando, posteriormente, as classes preditas no hiperplano.

ARAUJO (2017) é a versão melhorada do snoReport 1.0, ferramenta cuja fora utilizada em ARAUJO (2016) para a classificação de snoRNAs usando uma combinação de estruturas secundárias e ML. A aprimoração do snoReport contemplou novos recursos para os snoRNAs *box C/D e H/ACA box*, desenvolvendo uma técnica robusta na fase de treinamento da SVM (com dados recentes de organismos vertebrados e o refinamento dos parâmetros *C* e *gamma* na SVM), consumindo ainda mais bancos de dados para expandir a coleção anterior do snoReport. Para validar a sua serventia, houve diversos testes em organismos animais os quais mostraram um ótimo desempenho de classificação.

AOKI; SAKAKIBARA (2018) aborda a classificação de ncRNAs fundado nas redes neurais convolucionais. O treinamento é feito por representações distributivas de 4 nucleotídeos que derivaram com sucesso as matrizes de peso de posição em kernels aprendidos que correspondem a sequência de *motifs* como locais de ligação a proteínas. A classificação de um par alinhado de duas sequências em classes positivas e negativas corresponde ao agrupamento das sequências de entrada. Depois de combinarmos a distribuição representativa de nucleotídeos de RNA com a informação da estrutura secundária específica para ncRNAs e ainda com perfis de mapeamento de leituras de sequência de próxima geração, o treinamento de CNNs para classificação de alinhamentos de sequências de RNA rendeu agrupamento preciso em termos de famílias ncRNA e superou os métodos de agrupamento existentes tradicionais para sequências de ncRNA. Interessantes sequências de *motifs* e estruturas secundárias conhecidas pelas famílias de snoRNAs, microRNA e tRNAs foram identificadas no estudo.

NAVARIN; COSTA (2017) sugere um estudo voltado a classificação funcional de ncRNAs fundamentado na implementação de um grafo kernel. Para lidar com entidades representadas como grafos, uma variedade de kernels têm sido propostos na literatura. Diferentes noções de similaridade são obtidas escolhendo diversos tipos de subestruturas a serem consideradas, desde caminhos até pequenos subgrafos. Existem várias maneiras de representar estruturas secundárias de RNA, incluindo as representações entre colchetes (onde os nucleotídeos são convertidos em nós e ligações em arestas) e representações em árvore (onde pares de bases são convertidos em nós 'tronco' e nucleotídeos de alça são convertidos em nós de 'loop'). Cada representaçãotem diferentes vantagens e desvantagens, incluindo perda de informações e complexidade de cálculo. A estratégia NSPDK, assim como no trabalho de ARAUJO (2017), é adotada com o objetivo de materializar a codificação de características implícitas que é chave para obter eficiência linear na fase de classificação. Neste artigo, a representação escolhida é a *loss-less*, ou seja, **sem perda**, onde os nós representam nucleotídeos e as arestas são as ligações entre eles, seja do tipo *backbone* ou do tipo de *encadernação*.

3.5 Conclusão dos resultados apresentados na análise

Através da análise e discussão dos resultados da revisão, percebemos que existe a exploração de modelos preditivos matemáticos para a classificação de ncRNAs em oposição

aos modelos tradicionais biológicos. Este fato deve-se pela alta taxa de *F-score*, em outras palavras, da acurácia no classificamento de classes para os ncRNAs. Apesar da perspectiva biológica e híbrida, em contraste com a matemática, sua escolha varia de acordo com o objeto de estudo analisado e a sua eficiência de identificação. Há algoritmos que são melhores para classificação de móleculas de DNAs, assim como há outros mecanismos de classificação que produzem resultados significativos para as moléculas de RNAs. No atual contexto, os projetos científicos revelam que a extração de características por um cenário matemático demonstrou ser relevante para classificação de ncRNAs. O principal enfoque da monografia é demonstrar o custo do algoritmo, o *pipeline* das etapas a serem executadas desde a entrada, a parametrização, treinamento e testes até a saída. Mesmo que muito progresso já tenha sido feito, existem incógnitas para este grupo importante de moléculas que podem ser respondidas pelo avanço do AM. Com base nas provas de conceito observadas é possível perceber a capacidade do modelo preditivo matemático de identificar os ncRNAs e do benefício da identificação em um âmbito biomedicinal.

4

Projeto

O capítulo de projeto disserta sobre os estágios do AM voltado ao campo de pesquisa abragendo detalhadamente a construção do conjunto de dados ponderando sobre a quantidade de sequências encontradas de cada classe de snoRNAs e as tomadas de decisão na construção do conjunto negativo.

Explicará a etapa de pré-processamento dando ênfase, sobretudo, aos cálculos estatísticos do percentil dos dados, a média, variância e o valor máximo e mínimo de sequências por família identificado são informações que irão compôr este capítulo. Esta fase mostra como é feito o balanceamento dos conjuntos positivos e negativos na elaboração do conjunto final a ser usado como entrada para o algoritmo de extração.

Na fase de extração introduzirá o conceito de automatização de *scripts* no que diz respeito aos algoritmos de extração empregues, aplicando o paralelismo de execução com o propósito de acelerar a extração de características do conjunto de dados.

Em seguida, explicará o processo de treinamento responsável em dividir o nosso conjunto de dados para treino e testes. Nesta seção haverá a menção a validação cruzada, o método *hold-out* de particionamento de dados, o algoritmo de classificação *Random Forest*, os hiperparâmetros do estimador sobre uma grade de parâmetros e a matriz de confusão que exibe a distribuição dos registros em termos de sua classe.

Por último, é esclarecido o *pipeline* de treinamento e os estudos de caso que norteará a pesquisa. As métricas são fundamentais nesta etapa para avaliar a acurácia do preditor em classificar as duas classes de snoRNAs (C/D e H/ACA).

4.1 Coleta de dados

A busca por sequências das duas classes de snoRNAs foi feita a partir do banco de dados **RFAM** no intuito de agrupar o conjunto de sequências por sua respectiva família.

Criou-se um arquivo de *script* em *shell* para baixar automaticamente todas as sequências de cada família, definindo o nome do arquivo baseado no nome da familia com o formato da extensão *fasta*, cujo é a extensão padrão de representação de sequências de nucleotídeos. Cada arquivo foi designado à pasta com o nome da sua classe de snoRNA.

Em sua totalidade, foram obtidas 4877 sequências de snoRNAs *C/D box* entre 475 famílias e 2813 sequências de snoRNAs *H/ACA* box entre 283 famílias para o conjunto positivo de dados.

Em contra-partida, condicionou-se o esforço em construir o conjunto negativo consumindo as famílias RNase P, 5S rRNA e tRNA e algoritmos de embaralhamento, o que proporcionou o total de 4999 sequências, sendo 2433 sequências de RNAs não pertencentes à snoRNAs e 2566 sequências aleatórias.

É evidente que a desproporcionalidade entre a quantidade de sequências de um grupo ao outro pode afetar inteiramente a classificação do snoRNA na etapa de aprendizado de máquina e, portanto, no momento deve ser considerado dados brutos não-processados.

4.2 Pré-processamento de dados

Usando as famílias como base de cálculo e construção do conjunto positivo, extraiu-se o percentil de 85% dos dados, a média aritmética, a variância, e o valor máximo e mínimo das quantidades de sequências conforme expresso na tabela 4.1.

Tabela 4.1: Métricas de cálculo do conjunto positivo

Classe	**Sequências**	**Famílias**	**Percentil (0.85)**	**Média**	**Variância**	**Máximo**	**Mínimo**
C/D box	4877	475	6	4	2.5589	7	2
H/ACA box	2813	283	22	5	27884.03	76	2

As métricas de cálculo balancearam por meio da média aritmética a quantidade de sequências esperadas por família para que o algoritmo de máquina de aprendizagem as consuma em agrupamento equivalente. Dessa forma, ao prefedinir essa condição, sobraram 1553 sequências de *C/D box* e 1013 sequências de *H/ACA box* para composição do conjunto positivo de dados.

Em consonância, é necessário do conjunto negativo para treinar e testar o modelo de classificação a ser gerado, então, a elaboração do conjunto negativo teve como regra fundamental que 50% do conjunto seria criado por sequências geradas aleatórias por um algoritmo de embaralhamento à medida que a outra metade seria formada por sequências genéticas de RNAs tais como Ribonuclease (RNase) P, 5S RNA ribossómico (rRNA) e RNA transportador (tRNA), considerando que o tamanho máximo delimitado para o conjunto negativo seria três vezes maior que o conjunto positivo. Assim, adquiriu-se um total de 3166 sequências.

Tendo esta condição preestabelecida, obteve-se 1500 sequências geradas aleatórias e 1666 sequências constituídas pela mesclagem de RNase P, 5S rRNA e tRNA, totalizando 3166 sequências no conjunto negativo.

4.3 Extração

Os métodos de extração de características utilizados são de natureza matemática como o mapeamento númerico com as transformações de Fourier (Real, Z-curve), as entropias de Shannon e Tsallis e as redes complexas. Todos os algoritmos de extração de características podem ser extraídos do Github de BONIDIA et al. (2021a).

A criação de *scripts* na etapa de extração foi primordial para automação das atividades repetitivas no que tange à eficiência e rapidez pois facilitou a adequação de parâmetros para os algoritmos de extração, a organização de entrada e saída de dados em arquivos (principalmente àqueles que continham o formato fasta em sua extensão) e a execução paralela dos algoritmos para acelerar a fase de extração e agrupamento de dados.

A extração retornou um arquivo no formato *csv* abrangendo as colunas com as características encontradas em cada família pelos algoritmos. Vale ressaltar que estes dados são puramente contínuos, logo, é possível que haja valores infinitos e que não sejam numéricos. É relevante ter a ciência desta propriedade dos dados pois posteriormente haverá um tratamento em torno destes valores no estágio de pré-execução do classificador.

Para agrupar todos esses arquivos de formato *csv* à classe de snoRNA pertencente, usou-se as ferramentas "pré-embutidas"do sistema Linux de concatenação e manipulação do contéudo existente do arquivo como *cat*, *awk*, *grep*.

4.4 Treinamento

No processo de treinamento, em busca de um resultado satisfatório do algoritmo classificador por meio das extrações de características obtidas, estabeleceu-se um fluxo de trabalho (4.1) com o passo-a-passo das operações do algoritmo:

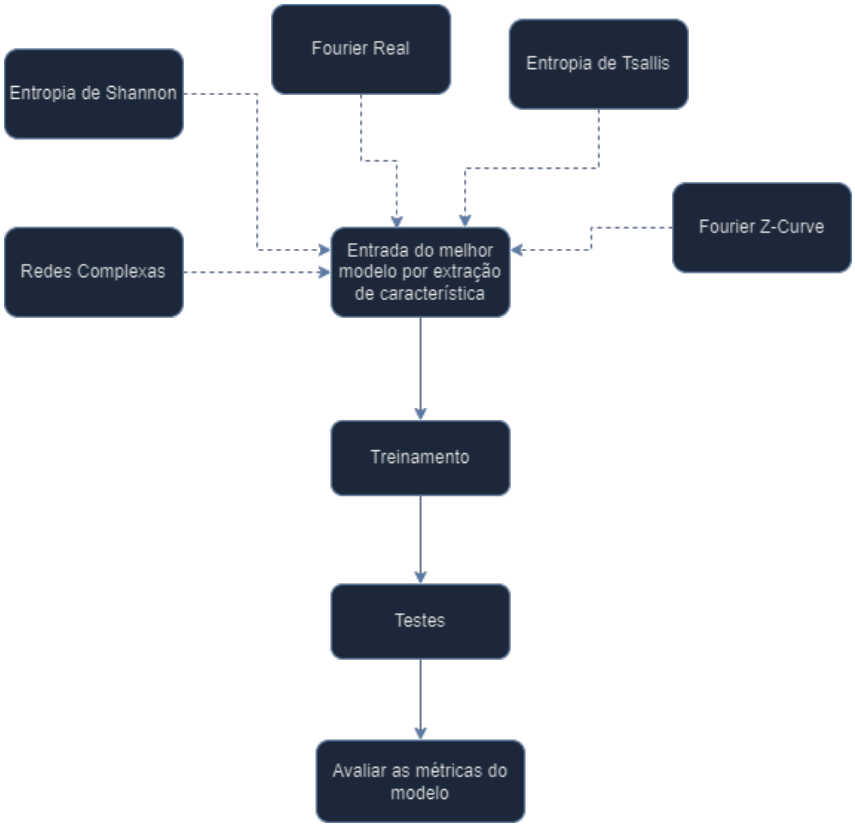


Figura 4.1: Fluxo de trabalho do algoritmo.

Dividiu-se o conjunto de treino e de testes tal que 70% do conjunto original ficou para treino enquanto os 30% restantes ficaram para o conjunto de testes e estes valores foram passados para a função **train_test_split** provida pelo pacote *sklearn.model_selection* em Python. No treinamento sem validação cruzada, há um parâmetro chamado **test_size** responsável por estabelecer a quantidade de iterações que o algoritmo de treino irá efetuar para que no final possa avaliar qual destes modelos de saída teve o melhor proveito. Em contrapartida, no treinamento com validação cruzada, o parâmetro **n_estimators** designa a proporção de modelos em uma única execução do algoritmo de modo que obtenha o melhor estimador entre a porção avaliada apoiado pelas métricas de avaliação.

Optou-se também em escolher o algoritmo de classificação *Random Forest* por ter sido um algoritmo promissor na pesquisa de BONIDIA et al. (2021a) a qual foi testada a sua generalização em diferentes tarefas de classificação para RNAs longos não-codificantes (lncRNAs) a partir de dados desbalanceados.

Os hiper-parâmetros de ajuste utilizados na Random Forest para cada método de extração de características estão dispostos na tabela 4.2.

Tabela 4.2: Hiperparâmetros da *Random Forest* sem usar a função *GridSearchCV*

Parâmetro	Valor
"bootstrap"	true
"ccp_alpha"	0.0
"class_weight"	None
"criterion"	gini
"max_depth"	10
"max_features"	sqrt
"max_leaf_nodes"	None
"max_samples"	None
"min_impurity_decrease"	0.0
"min_samples_leaf"	1
"min_samples_split"	2
"min_weight_fraction_leaf"	0.0
"n_estimators"	100
"n_jobs"	None
"oob_score"	false
"random_state"	None
"verbose"	0
"warm_start"	false

Para automatizar este processo de *tuning* de hiperparâmetros, foi-se utilizado a função *GridSearchCV* do módulo *sklearn* em Python. O objetivo primário do *GridSearchCV* é a criação de combinações de parâmetros a partir de uma busca exaustiva sobre valores especificados para um estimador (*score*, ou seja, métrica de avaliação), para posteriormente avaliá-las.

Os parâmetros do estimador usados para aplicar esses métodos são otimizados e refinados por validação cruzada (*cross-validation*) sobre uma grade de parâmetros.

De forma semelhante aos hiperparâmetros padrões da *Random Forest*, o *GridSearchCV* aplicou os seguintes parâmetros conforme mostrado na tabela 4.3.

Tabela 4.3: Hiperparâmetros da *Random Forest* após o uso da função *GridSearchCV*

Parâmetro	Valor
"mean_fit_time"	array([0.03470263, 0.34155726, 1.70107441])
"std_fit_time"	array([0.00415981, 0.02498759, 0.14168099])
"mean_score_time"	array([0.00217724, 0.01229601, 0.04444399])
"std_score_time"	array([0.0001098 , 0.00510206, 0.01017052])
"param_n_estimators"	masked_array(data=[10, 100, 500])
"mask"	array([False, False, False]
"params"	array['n_estimators': 10, 'n_estimators': 100, 'n_estimators': 500]
"split0_test_score"	array([0.98817967, 0.99061033, 0.99061033])
"split1_test_score"	array([0.98337292, 0.98329356, 0.98337292])
"split2_test_score"	array([0.98584906, 0.98352941, 0.98352941])
"split3_test_score"	array([0.98345154, 0.98113208, 0.98113208])
"split4_test_score"	array([0.98578199, 0.98584906, 0.98578199])
"mean_test_score"	array([0.98532703, 0.98488289, 0.98488535])
"std_test_score"	array([0.00178623, 0.00322997, 0.00321846])
"rank_test_score"	array([1, 3, 2])

4.5 Estudo de caso: classificação de snoRNAs em conjunto de dados encontrados na literatura

Nos estudos de casos, as operações foram divididas em **N** execuções e para cada execução ocorrerá a verificação das métricas de avaliação para que na etapa de testes seja escolhido o melhor modelo encontrado para cada método de extração.

As validações de treinamento envolvem qualquer validação em que o modelo precise ser retreinado. Normalmente, isso inclui testar diferentes modelos durante um único *pipeline* de treinamento. Essas validações são realizadas nesta fase de treinamento/avaliação do desenvolvimento do modelo, e muitas vezes são mantidas como código de experimentação, não fazendo parte do produto final do classificador.

O *pipeline* de treinamento inicia-se ao carregar o modelo preditivo com a melhor acurácia na pontuação *f1_score* por método de extração de característica, é então feito dois estudos de caso em torno do conjunto de dados do mundo real como o genoma de vertebrados e invertebrados tais como galinhas, moscas pertencentes à família *Drosophilidae*, nematódeos da família *Rhabditidae*, protozoários da família *Trypanosomatidae* como o leishmania, humanos e de ornitorrincos:

1. Estudo de Caso: Adicionar o conjunto de dados reais de acordo com sua respectiva classe de snoRNAs dos genomas encontrados e usar o modelo para predizer este conjunto.

2. Estudo de Caso: Comparar os resultados obtidos pela predição do conjunto de treinamento avaliando o comportamento do classificador com referências de outros artigos que predizeram as duas classes de snoRNAs (*C/D box* e *H/ACA box*)

Para cada estudo de caso, é efetuado um teste sem validação cruzada e outro com validação cruzada para fins comparativos entre a acurácia da predição por modelo.

Antes da estimação do modelo preditivo, nos estudos de caso em que é feito uma validação cruzada, a execução do treinamento divide o conjunto em dados de treino e de testes em diferentes partes do modelo de forma que valide o desempenho de cada modelo em um dado intervalo, garantindo a generalização dos dados apresentados dentre os melhores parâmetros encontrados.

O cálculo de acertos e erros é feito pela matriz de confusão que mostra as frequências de classificação para cada classe de snoRNAs. A matriz nos conduz a uma breve análise das estimativas ainda que não tenha sido englobada a uma métrica de avaliação, como a figura 4.2 exemplifica.

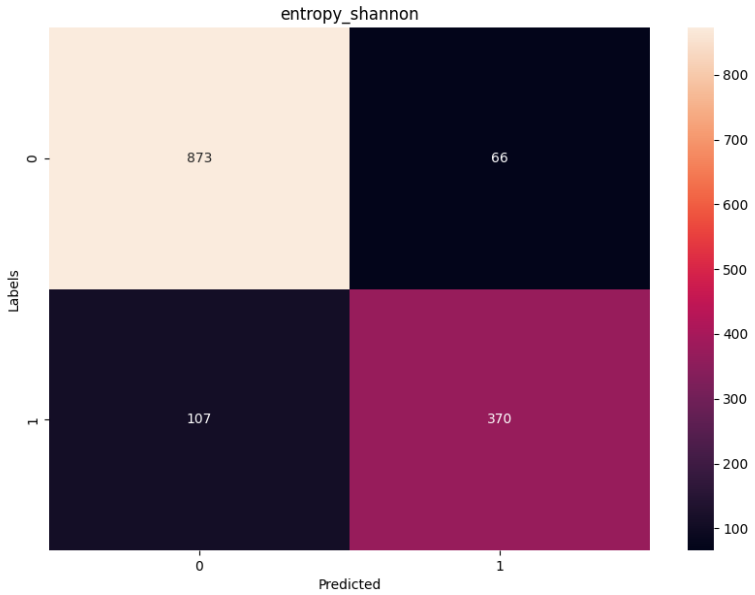


Figura 4.2: Matriz de confusão na etapa de treinamento usando o método de Entropia de Shannon para a classe de snoRNAs *C/D box*. (*Labels* são rótulos e *Predicted* são as predições.)

5

Resultados

Primeiro, será apresentado as estatísticas dos testes de desempenho. Então, posteriormente haverá os resultados da execução do classificador em dados reais para verificar a acurácia das métricas em diferentes organismos.

5.1 Estatísticas

Para identificar snoRNAs *H/ACA box* e *C/D*, construiu-se dois conjuntos de dados diferentes para cada classe de snoRNAs. Para as fases de aprendizado, usou-se um conjunto de dados como treinamento e o outro para teste usando o método *hold-out* de separação e de validação cruzada respectivamente. Cada treinamento foi repetido 10 vezes, e o cálculo das métricas, desvio padrão e a média estão dispostos nas tabelas para cada classe de snoRNA e método de extração utilizado. É de suma importância saber que estas métricas foram extraídas dos melhores estimadores, ou seja, do melhor modelo encontrado baseado pelo **f1_score** em torno dos treinamentos envolvidos.

Tabela 5.1: Resultados da fase de teste para snoRNAs C/D box: F-score (FSC), Acurácia (Acc), *Recall* (REC), Precisão Média (PRE), Área sob a curva ROC (AUC). A média e desvio padrão total de cada métrica

Classe (snoRNAs)	Método de extração	FSC(%)	ACC(%)	REC(%)	PRE(%)	AUC(%)
C/D box	Fourier Real	98.25	98.81	97.26	99.18	99.85
C/D box	Fourier Z-Curve	98.81	99.15	98.27	99.35	99.96
C/D box	Entropia de Shannon	79.83	87.37	76.47	84.01	93.71
C/D box	Entropia de Tsallis	79.34	86.58	78.34	80.09	93.35
C/D box	Redes Complexas	99.72	99.79	99.53	99.94	99.98
Média (%)		90,70	94.35	89.97	92.51	97.37
Desvio padrão (%)		10,60	6,73	11,52	9,65	2,72

Tabela 5.2: Resultados da fase de teste para snoRNAs H/ACA box: F-score (FSC), Acurácia (Acc), *Recall* (REC), Precisão Média (PRE), Área sob a curva ROC (AUC).

Classe (snoRNAs)	Método de extração	FSC(%)	ACC(%)	REC(%)	PRE(%)	AUC(%)
H/ACA box	Fourier Real	95.85	98.01	94.08	98.28	98.92
H/ACA box	Fourier Z-Curve	96.83	98.49	95.88	98.12	99.46
H/ACA box	Entropia de Shannon	40.69	77.84	31.69	58.61	78.92
H/ACA box	Entropia de Tsallis	50.41	80.20	44.41	63.21	84.81
H/ACA box	Redes Complexas	97.88	98.97	96.41	99.55	99.83
Média (%)		76,33	90,70	72,49	83,55	92,38
Desvio padrão (%)		28,31	10,70	31,77	20,74	9,83

5.2 Estudo de caso

Para validar o classificador Random Forest a partir de métodos matemáticos, usou-se os conceitos de validação cruzada para separar cada k-dobra ou k-parte em que o valor de k=5. Para cada dobra, separou-se ambos conjuntos e calculou-se as métricas desejadas (F1, AUC, PRE, REC, ACC). De acordo com o melhor estimador o modelo foi escolhido e separado para que seja avaliado em um conjunto de dados reais com sequências previstas de vertebrados e invertebrados, alguns desses organismos foram parcialmente confirmados em experimentos anteriores sejam em humanos, nematóides, drosofilídeos, ornitorrincos, galinhas e leishmania. Todas essas sequências foram extraídas do artigo ARAUJO (2017) que fora consumido como base comparativa de resultados.

Em ARAUJO (2017), o autor consumiu os conjuntos de validação dos artigos referidos na listagem abaixo usando o snoReport 2.0, software do qual usa previsão de estrutura secundária de RNA para identificar as duas classes de snoRNAs (*H/ACA box*) e (*C/D box*):

1. YANG et al. (2006) que usa o snoSeeker, um método baseado em modelos probabilísticos, emparelhamento de segmentos do genoma, para encontrar snoRNAs em equências genéticas de *Homo Sapiens*.

2. Do trabalho de SCHMITZ et al. (2008) que busca as sequências genéticas em biblioteca de **cDNA** (combinação de fragmentos clonados de cDNA armazenados em células hospedeiras) do cérebro do ornitorrinco, gerado a partir de pequenos RNAs não codificadores de proteína.

3. Do artigo científico de ZEMANN et al. (2006), que extrai sequências genéticas do conjunto de *Caernorhabditis elegans*, isto é, um invertebrado da espécie Nematódeo

para ser usado na combinação de triagem de biblioteca de cDNA com estratégias de busca e análise computacional para encontrar snoRNAs.

4. De HUANG et al. (2005) o qual perfoma uma análise de larga escala em *Drosophila melanogaster* usando métodos *RNomics* experimentais e computacionais para identificar classes de snoRNAs.

5. Por fim, o artigo de LIANG et al. (2007) que utiliza patógenos intimamente relacionados de *Leishmania major* para identificar classes de snoRNAs a partir da abordagem de *screening* por todo o genoma.

A tabela 5.3 mostra os resultados obtidos pelo snoReport 2.0 de ARAUJO (2017).

Tabela 5.3: Resultados do snoReport nas classes de snoRNAs (*C/D box* e *H/ACA box*).

Conjunto		
Homo Sapiens	C/D: (21/21)	H/ACA: (28/32)
Platypus	C/D: (42/144)	H/ACA: (45/73)
Gallus gallus	C/D: (122/132)	H/ACA: (66/69)
Nematodes	C/D: (32/108)	H/ACA: (46/60)
Drosophila	C/D: (2/63)	H/ACA: (39/56)
Leishmania	C/D: (0/62)	H/ACA: (0/37)

Comparando os resultados obtidos em ARAUJO (2017) pelo snoReport 2.0 levando em conta os conjuntos de validação dos artigos citados, as tabelas 5.10 mostram o quão eficaz foi o preditor em classificar os snoRNAs em *C/D box* ou *H/ACA box* tendo como base comparativa o trabalho mencionado conforme tabela 5.4:

Tabela 5.4: Resultados obtidos no trabalho de ARAUJO (2017) usando o software snoReport 2.0 nas classes de snoRNAs.

Conjunto	C/D	H/ACA
Homo Sapiens	(21/21)	(28/32)
Platypus	(42/144)	(45/73)
Gallus gallus	(112/132)	(66/69)
Nematodes	(32/108)	(46/60)
Drosophila	(2/63)	(39/56)
Leishmania	(0/62)	(0/37)

Tabela 5.5: Resultados obtidos em nosso classificador usando os métodos de extração de teor matemático nas classes de snoRNAs.

Tabela 5.6: Método de Fourier Real

Conjunto	C/D	H/ACA
Homo Sapiens	(21/21)	(28/33)
Platypus	(143/144)	(69/73)
Gallus gallus	(124/132)	(67/69)
Nematodes	(106/108)	(60/60)
Drosophila	(63/63)	(55/56)
Leishmania	(54/62)	(36/37)

Tabela 5.7: Método de Fourier Z-Curve

Conjunto	C/D	H/ACA
Homo Sapiens	(21/21)	(33/33)
Platypus	(144/144)	(72/73)
Gallus gallus	(125/132)	(69/69)
Nematodes	(106/108)	(59/60)
Drosophila	(63/63)	(55/56)
Leishmania	(61/62)	(36/37)

Tabela 5.8: Método de Entropia de Shannon

Conjunto	C/D	H/ACA
Homo Sapiens	(21/21)	(18/33)
Platypus	(128/144)	(30/73)
Gallus gallus	(109/132)	(24/69)
Nematodes	(73/108)	(13/60)
Drosophila	(46/63)	(8/56)
Leishmania	(45/62)	(17/37)

Tabela 5.9: Método de Entropia de Tsallis

Conjunto	C/D	H/ACA
Homo Sapiens	(19/21)	(27/33)
Platypus	(127/144)	(52/73)
Gallus gallus	(114/132)	(24/69)
Nematodes	(83/108)	(17/60)
Drosophila	(49/63)	(30/56)
Leishmania	(54/62)	(21/37)

Tabela 5.10: Método de Redes Complexas.

Conjunto	C/D	H/ACA
Homo Sapiens	(21/21)	(33/33)
Platypus	(144/144)	(73/73)
Gallus gallus	(127/132)	(69/69)
Nematodes	(107/108)	(60/60)
Drosophila	(62/63)	(56/56)
Leishmania	(61/62)	(37/37)

Usando o método da Transformação de Fourier com o mapeamento numérico de representação **Real** na classe de snoRNAs *C/D box*, 96.79% das sequências foram encontradas. Na

classe de snoRNAs *H/ACA box*, o classificador encontrou 94.81% das sequências.

Com o método da Transformação de Fourier com o mapeamento numérico de representação **Z-curve** na classe de snoRNAs *C/D box*, 98.49% das sequências foram encontradas. Na classe de snoRNAs *H/ACA box*, 98.78% das sequências foram encontradas.

Já com o método de **Entropia de Shannon** na classe de snoRNAs *C/D box*, 76.79% das sequências foram encontradas, tendo essa defasagem na estimativa em comparação com os métodos de Fourier. Na classe de snoRNAs *H/ACA box*, apenas 35.67% das sequências foram encontradas, tendo o pior resultado entre os outros métodos relatados.

Na **Entropia de Tsallis**, o método conseguiu encontrar 82.26% das sequências de organismos vertebrados e invertebrados para a classe de snoRNAs *C/D box*. Na classe de snoRNAs *H/ACA box*, 58.84% das sequências foram encontradas.

Por fim, o método de **Redes Complexas** na classe de snoRNAs *C/D box* encontrou 98.49% das sequências. Na classe de snoRNAs *H/ACA box*, 99.69% das sequências foram encontradas.

Separadamente, em resumo, o classificador foi eficiente em identificar os organismos vertebrados e invertebrados do conjunto de validação. Considerando que o total de sequências da classe snoRNAs *C/D box* de organismos vertebrados é **297** e invertebrados é **233** e para a da classe snoRNAs *H/ACA box* é **175** e **153** respectivamente, é evidente que os algoritmos de Fourier e o de Redes Complexas foram consideravelmente significativos na classificação tendo uma acurácia maior que 90% em ambas predições das duas classes de snoRNAs. Ainda que os métodos de Entropia não foram tão eficientes, para a classe snoRNAs C/D box conseguiram ter uma eficiência em torno dos 80% de acurácia,

As tabelas 5.12 e 5.13 mostram o diagnóstico da quantidade de sequências encontradas por método de extração de características em organismos vertebrados e invertebrados em comparação com a tabela 5.11.

Tabela 5.11: Quantidade de sequências encontradas usando o snoReport 2.0 no trabalho de ARAUJO (2017).

Ferramenta	C/D	Acurácia (C/D)	H/ACA	Acurácia (H/ACA)
*s*noReport 2.0 (vertebrados)	175	58.92	139	79.88
*s*noReport 2.0 (invertebrados)	34	14.59	85	55.55

Tabela 5.12: Quantidade de sequências encontradas por método de extração de características em organismos vertebrados

Método	C/D	Acurácia (C/D)	H/ACA	Acurácia (H/ACA)
Fourier Real	288	96.96	161	96.56
Fourier Z-Curve	291	97.97	174	99.14
Entropia de Shannon	250	84.17	84	67.38
Entropia de Tsallis	255	85.85	128	77.68
Redes Complexas	292	98.31	175	98.71

Tabela 5.13: Quantidade de sequências encontradas por método de extração de características em organismos invertebrados

Método	C/D	Acurácia (C/D)	H/ACA	Acurácia (H/ACA)
Fourier Real	225	92.0	150	98.03
Fourier Z-Curve	231	99.42	150	98.03
Entropia de Shannon	157	48.0	65	42.48
Entropia de Tsallis	181	73.14	88	57.51
Redes Complexas	230	100.0	152	99.34

5.3 Discussão

Neste trabalho, construiu-se um conjunto de dados pré-processado usando uma estratégia de filtração de dados no intuito de remover a redundância entre eles, além de implementar um algoritmo de embralhamento do sequencial genético para produzir o conjunto negativo. Refinou-se a fase de treinamento do método *Random Forest* escolhendo cuidadosamente os parâmetros do classificador *Random Forest* usando pesquisas de grade (função *GridSearchCV*). Diferentes métodos de extração de características de cunho matemático foram aplicados com a finalidade de verificar a consistência dos algoritmos na classificação de snoRNAs. Diversas métricas de avaliação foram inclusas para apreciação de acurácia do modelo preditivo de forma geral. Estes procedimentos permitem avaliar a perfomance do classificador ao longo do processo excessivo e extenso de aprendizado de máquina os quais configuram as estimativas de snoRNAs no modelo pré-estabelecido de entrada.

Percebe-se que nos resultados alcançados, os algoritmos de extração de Fourier com o mapeamento numérico Real e Z-curve tiveram um destaque atingindo uma proporção maior que

90% em todas as medidas de desempenho apresentadas tanto para snoRNAs *C/D box* quanto para snoRNAs *H/ACA box* permitindo-nos ter alta taxa de qualidade em cada previsão. Assim como os algoritmos de Fourier tiveram tal atenção, o algoritmo de Redes Complexas atingiu uma taxa similar em consonância com os resultados de Fourier, demonstrando serem ótimos métodos de extração de característica para o algoritmo *Random Forest*.

Embora os algoritmos citados tenham sido efetivos na classificação, os de natureza entrópica não foram efetivos na predição atingindo um F-score em torno de 80% na predição de snoRNAs *C/D box* e aproximadamente 45% na predição de snoRNAs *H/ACA box*. Embora a curva ROC AUC apresente um alto índice de confiabilidade exibindo uma porcentagem quase maior que 80%, ainda assim o algoritmo mostrou ser insuficiente em predizer as amostras.

Vale ressaltar que, segundo ARAUJO (2017), muitas sequências usadas para validação ainda não foi experimentalmente validadas, e talvez alguns deles possam ser falsos positivos, ou não são representantes dos snoRNAs canônicos (como os snoRNAs na leishmânia).

Outra observação importante é que a validação envolveu apenas as amostras positivas de dados e portanto é crucial fazer uma avaliação negativa envolvendo também o conjunto negativo para que o algoritmo de classificação possa ter mais trabalho em classificar os organismos vertebrados e invertebrados.

Apesar disso, ainda que a fase de treinamento não tenha sido composto pelo conjunto negativo, o classificador identificou 96.60% snoRNAs *C/D box* e 96.03% snoRNAs *H/ACA box* nos organismos vertebrados e invertebrados usando o método de Fourier Real enquanto o método de Fourier Z-Curve encontrou 98.49% snoRNAs *C/D box* e 98.78% snoRNAs *H/ACA box*. O algoritmo de Redes Complexas identificou 98.49% snoRNAs *C/D box* e 99.69% snoRNAs *H/ACA box* nos organismos vertebrados e invertebrados.

Portanto, o classificador é eficiente nos métodos de extração de Fourier e Redes Complexas para identificar ambas as classes de snoRNAs e pode ser usado para organismos diferentes, sendo vertebrados ou invertebrados, com alta qualidade de previsão.

6

Conclusão

Essa metodologia adotada pelo modelo de AM em modificar e rotular a informação não estruturada em valores contínuos e discretos contribui para compreensão e visualização de cada aspecto presente, mesmo que não tão "visível", das estruturas genômicas espalhadas em organismos vivos. Na proporção em que o avanço científico na área de Inteligência Artifical vêm ganhando forças, novas abordagens são constatadas no meio em torno dos processos de extrações de características, sejam eles de natureza biológica, matemática e/ou híbrida. É expressivo e relevante compreender como cada algoritmo de extração se comporta diante a classificação binária de sequências genômicas para que abra espaço para elucidação do genoma em um espectro científico que irá proporcionar descobertas de novas doenças, novos padrões de proteínas e na criação de remédios contra estas novas doenças. Entender um padrão e correlacioná-lo à uma classe, ou melhor, conseguir categorizar um grupo no mundo biológico não é uma tarefa simples, tanto que diferentes abordagens são disseminadas porém, nem todas são eficazes o suficiente para serem relevantes.

O estudo de novos métodos, principalmente os de caráter matemático, demonstrou ser promissor diante da constatação e validação dos experimentos apresentados em torno destes algoritmos de teor matemático. Em particular, o algoritmo de transformação numérica de Fourier e o algoritmo de Redes Complexas que revelaram uma taxa significativa nas métricas de avaliação tornando-se expressivos na classificação de pequenos RNAs nucleolares (snoRNAs), em exclusivo às classes *C/D box* e *H/ACA box* na medida em que reconhece o conjunto e adquire conhecimento sobre as *features* no processo preditivo. Ainda que solidificado em um único algoritmo de classificação (*Random Forest*), os algoritmos de extração provaram serem capazes de traduzir a informação não descritiva das sequências genéticas em uma amostragem descricionária sobre os snoRNAs para que o classificador possa designar cada característica ás sub-árvores do *Random Forest* e aperfeiçoá-las com seus hiperparâmetros. No fim, as suas tomadas de decisão sucedeu um percentual maior que 90% de acurácia no estágio de validação do conjunto real de dados.

Para trabalhos futuros, uma ideia seria incluir novos conjuntos de dados para diferentes tipos de organismos (não apenas aqueles que foram citados nos resultados) e comparar as métricas do classificador *Random Forest* usando diferentes métodos de aprendizado de máquina (por

exemplo, SVM, *K-Nearest Neighbor* (KNN) ou até mesmo o EDeN que é baseado no conceito do KNN mas para grafos) pois pode ser que encontrem características intrínsecas ou mesmo prever novos snoRNAs. Poderia também treinar os dados utilizando outros hiperparâmetros para os classificadores, sempre analisando a melhor abordagem para àquele determinado conjunto e evitando o *overfitting*. Outra sugestão é se beneficiar do classificador para a construção de uma ferramenta de identificação de snoRNAs em genomas. (identificação seria encontrar os candidados a snoRNAs no genoma e usar o classificador logo em seguida). Além disso, como mencionado no tópico de *Discussão*, é essencial fazer uma avaliação negativa envolvendo também amostras negativas no conjunto de validação para que o algoritmo de classificação possa ter mais trabalho em classificar os organismos vertebrados e invertebrados.

Referências

AL-MASRI, A. **How Does Back-Propagation in Artificial Neural Networks Work?** 2019.

ALLALI, E. et al. Machine learning applications in RNA modification sites prediction. **Computational and Structural Biotechnology Journal**, [S.l.], v.19, p.5510–5524, 2021.

AOKI, G.; SAKAKIBARA, Y. Convolutional neural networks for classification of alignments of non-coding RNA sequences. **Bioinformatics**, [S.l.], v.34, n.13, p.i237–i244, 06 2018.

ARAUJO, J. V. de. Identificação de snoRNAs usando aprendizagem de máquina. **Bioinformatics**, [S.l.], p.1–105, 06 2016.

ARAUJO, J. V. de. SnoReport 2.0: new features and a refined support vector machine to improve snorna identification. **BMC Bioinformatics**, [S.l.], p.1–14, 11 2017.

BONIDIA et al. Feature extraction approaches for biological sequences: a comparative study of mathematical features. **Briefings in Bioinformatics**, [S.l.], v.22, n.5, 02 2021.

BONIDIA et al. MathFeature: feature extraction package for dna, rna and protein sequences based on mathematical descriptors. **Briefings in Bioinformatics**, [S.l.], v.23, n.1, 11 2021. bbab434.

CALDARELLI, G. **Scale-Free Networks: complex webs in nature and technology**. [S.l.]: Oxford University Press, 2007.

CHAKRAVORTY, N. Non-coding RNAs: the silent regulators of health and diseases. In: REPORTS, M. B. (Ed.). . [S.l.]: SpringerLink, 2022.

CHATTERJEE, S. **What is Feature Extraction? Feature Extraction in Image Processing**. 2022.

CHEN, W. et al. iRNA-Methyl: identifying n6-methyladenosine sites using pseudo nucleotide composition. **Analytical Biochemistry**, [S.l.], v.490, p.26–33, 2015.

COSTA, F.; GRAVE, K. D. Fast Neighborhood Subgraph Pairwise Distance Kernel. In: INTERNATIONAL CONFERENCE ON INTERNATIONAL CONFERENCE ON MACHINE LEARNING, 27., Madison, WI, USA. **Proceedings...** Omnipress, 2010. p.255–262. (ICML'10).

COURONNÉ et al. Random forest versus logistic regression: a large-scale benchmark experiment. **BMC Bioinformatics**, England, v.19, n.1, p.270, July 2018.

DEMŠAR, J.; ZUPAN, B. Hands-on training about overfitting. **PLoS Comput. Biol.**, [S.l.], v.17, n.3, p.e1008671, Mar. 2021.

DERNONCOURT, F. **Interpreting the AUROC**. 2015.

DIECI, G. et al. Eukaryotic snoRNAs: a paradigm for gene expression flexibility. **Genomics**, [S.l.], v.94, n.2, p.83–88, 2009.

DöRING, M. **Interpreting ROC Curves, Precision-Recall Curves, and AUCs**. 2018.

GOLDSTEIN-GREENWOOD, J. **ROC Curves and AUC for Models Used for Binary Classification**. 2022.

GUO, Y. **Cross-Validation and Hyperparameter Tuning**. 2021.

GUSIC, M.; PROKISCH, H. ncRNAs: new players in mitochondrial health and disease? In: GENET, F. (Ed.). . [S.l.]: frontiers in Genetics, 2020.

HENNIG, M. et al. Introduction of a time series machine learning methodology for the application in a production system. **Advanced Engineering Informatics**, [S.l.], v.47, p.101197, 2021.

HEYNE, S. et al. GraphClust: alignment-free structural clustering of local rna secondary structures. **Bioinformatics**, [S.l.], v.28, n.12, p.i224–i232, 06 2012.

HUANG, Z. hao et al. snoRNAs: functions and mechanisms in biological processes, and roles in tumor pathophysiology. In: DISCOVERY, C. D. (Ed.). . [S.l.]: Nature, 2022.

HUANG, Z.-P. et al. Genome-wide analyses of two families of snoRNA genes from Drosophila melanogaster, demonstrating the extensive utilization of introns for coding of snoRNAs. **RNA**, [S.l.], v.11, n.8, p.1303–1316, Aug. 2005.

KARACA, Y.; MOONIS, M. Chapter 14 - Shannon entropy-based complexity quantification of nonlinear stochastic process: diagnostic and predictive spatiotemporal uncertainty of multiple sclerosis subgroups. In: KARACA, Y. et al. (Ed.). **Multi-Chaos, Fractal and Multi-Fractional Artificial Intelligence of Different Complex Systems**. [S.l.]: Academic Press, 2022. p.231–245.

KITCHENHAM, B.; BRERETON, P. A systematic review of systematic review process research in software engineering. **Information and Software Technology**, [S.l.], v.55, n.12, p.2049–2075, 2013.

KOSLICKI, D. Topological entropy of DNA sequences. **Bioinformatics**, [S.l.], v.27, n.8, p.1061–1067, 02 2011.

LIANG, X.-H. et al. Genome-wide analysis of C/D and H/ACA-like Small nucleolar RNAs in *Leishmania major* indicates conservation among trypanosomatids in the repertoire and in their rRNA targets. **Eukaryot. Cell**, [S.l.], v.6, n.3, p.361–377, Mar. 2007.

LIMA, A. M.; PORTILLO, H. A. del. Computational methods in noncoding RNA research. In: MATHEMATICAL BIOLOGY, J. of (Ed.). . [S.l.]: SpringerLink, 2007.

LYASHENKO, V. **Cross-Validation in Machine Learning**: how to do it right. 2023.

MALADKAR, K. **Overview Of Convolutional Neural Network In Image Classification**. 2018.

MATA, A. S. d. Complex Networks: a mini-review. **Brazilian Journal of Physics**, [S.l.], v.50, n.5, p.658–672, Oct 2020.

MENDIZABAL-RUIZ, G. et al. On DNA numerical representations for genomic similarity computation. **PLoS One**, United States, v.12, n.3, p.e0173288, Mar. 2017.

MENDIZABAL-RUIZ, G. et al. On DNA numerical representations for genomic similarity computation. **PLoS One**, United States, v.12, n.3, p.e0173288, Mar. 2017.

MITCHELL, T. M. **Machine Learning**. 1.ed. USA: McGraw-Hill, Inc., 1997.

MüLLER, A. C. **Data Splitting Strategies**. 2020.

NAVARIN, N.; COSTA, F. An efficient graph kernel method for non-coding RNA functional prediction. **Bioinformatics**, [S.l.], v.33, n.17, p.2642–2650, 05 2017.

PANT, A. **Workflow of a Machine Learning project**. 2019.

PAYNE, S. Chapter 3 - Virus Interactions With the Cell. In: PAYNE, S. (Ed.). **Viruses**. [S.l.]: Academic Press, 2017. p.23–35.

PRAMODITHA, R. **Addressing Overfitting 2023 Guide — 13 Methods**. 2022.

PRUSTY, S.; PATNAIK, S.; DASH, S. K. SKCV: stratified k-fold cross-validation on ml classifiers for predicting cervical cancer. **Frontiers in Nanotechnology**, [S.l.], v.4, 2022.

RABELLO, E. B. **Cross Validation**: avaliando seu modelo de machine learning. 2019.

RANA, J.; VAISLA, D. K. **Introduction To Bioinformatics**. [S.l.: s.n.], 2012. p.11–18.

SANTOS, V. S. dos. **Aminoácidos**. 2019.

SCHADE, G. **Azure Machine Learning**. 2018.

SCHMITZ, J. et al. Retroposed SNOfall–a mammalian-wide comparison of platypus snoRNAs. **Genome Res.**, [S.l.], v.18, n.6, p.1005–1010, June 2008.

SCHONLAU, M.; ZOU, R. Y. The random forest algorithm for statistical learning. **The Stata Journal**, [S.l.], v.20, n.1, p.3–29, 2020.

SEMMLOW, J. Chapter 3 - Fourier Transform: introduction. In: SEMMLOW, J. (Ed.). **Signals and Systems for Bioengineers (Second Edition)**. Second Edition.ed. Boston: Academic Press, 2012. p.81–129. (Biomedical Engineering).

SETUBAL, J. C.; MEIDANIS, J. Introduction to computational molecular biology. In: OMNIPRESS. **Anais...** [S.l.: s.n.], 1997.

SHAIKH, R. **Cross Validation Explained**: evaluating estimator performance. 2018.

SZCZEÅNIAK, M. W. et al. Towards a deeper annotation of human lncRNAs. **Biochimica et biophysica acta. Gene regulatory mechanisms**, [S.l.], v.1863, n.4, p.194385, April 2020.

TORRES-GARCíA, A. A. et al. Chapter 4 - Pre-processing and feature extraction. In: TORRES-GARCíA, A. A. et al. (Ed.). **Biosignal Processing and Classification Using Computational Learning and Intelligence**. [S.l.]: Academic Press, 2022. p.59–91.

VIEIRA, L. M. et al. PlantRNA$_{S}niffer$: $asvm-basedworkflowtopredictlongintergenicnon-codingrnasinplants$.**Non-Coding RNA**, $[S.l.], v.3, n.1, 2017$.

WANG, Y. et al. Computational identification of human long intergenic non-coding RNAs using a GA–SVM algorithm. **Gene**, [S.l.], v.533, n.1, p.94–99, 2014.

GARGAUD, M. et al. (Ed.). **Watson–Crick Pairing**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011. p.1775–1776.

WURTZ, R. Recounting the impact of Hubel and Wiesel. **The Journal of Physiology**, [S.l.], v.587, 2009.

YANG, J.-H. et al. snoSeeker: an advanced computational package for screening of guide and orphan snoRNA genes in the human genome. **Nucleic Acids Res**, England, v.34, n.18, p.5112–5123, Sept. 2006.

YIN, C.; YAU, S. S.-T. A Fourier characteristic of coding sequences: origins and a non-fourier approximation. **J Comput Biol**, United States, v.12, n.9, p.1153–1165, Nov. 2005.

ZACHARY, W. W. An Information Flow Model for Conflict and Fission in Small Groups. **Journal of Anthropological Research**, [S.l.], v.33, n.4, p.452–473, 1977.

ZEMANN, A. et al. Evolution of small nucleolar RNAs in nematodes. **Nucleic Acids Res.**, [S.l.], v.34, n.9, p.2676–2685, May 2006.

ZHANG, J. Dive into Decision Trees and Forests: a theoretical demonstration. , [S.l.], p.44, 01 2021.