UNIVERSIDADE FEDERAL DE SÃO CARLOS

THAYANA VIEIRA TAVARES

APRENDIZADO DE MÁQUINA PARA CLASSIFICAÇÃO DAS LINHAGENS DE COPIA E GYPSY DE ANGIOSPERMAS

SÃO CARLOS - SP
2021

THAYANA VIEIRA TAVARES

APRENDIZADO DE MÁQUINA PARA CLASSIFICAÇÃO DAS LINHAGENS DE COPIA E GYPSY DE ANGIOSPERMAS

Trabalho apresentado como requisito para a obtenção
do título de Bacharel em Biotecnologia
pela Universidade Federal de São Carlos.
Orientador: Prof. Dr. Ricardo Cerri.

SÃO CARLOS - SP
2021

Agradecimentos

À meus familiares que sempre apoiaram todas as minhas escolhas nos estudos, inclusive a troca de estado para realizar uma graduação. Desses agradecimentos, boa parte é para minha vó, sem ela, chegar até aqui teria sido impossível.

Aos meus melhores amigos de sala de aula, que foram um enorme alicerce em dias tão difíces. Em tempos nebulosos e cinzas, eles foram arco-íris.

À toda a comunidade PyLadies, mas principalmente ao capítulo PyLadies São Carlos, por todo o suporte e por ter feito minha jornada na aprendizagem de computação tão leve e prazerosa. Além do acolhimento, recebi impulso.

Ao Prof. Dr. Ricardo Cerri, por todas as dicas, orientações e pela paciência de acolher uma estudante de biológicas encantada com a computação em um projeto inteiramente voltado ao aprendizado de máquina.

Admirar e aproveitar o caminho é mais importante do que chegar,
porque a vida não é uma corrida.

Resumo

Este estudo utilizou algoritmos de aprendizado de máquina (rede neural, árvore de decisão e algoritmo dos vizinhos próximos) para criar modelos de classificação de 11 linhagens das superfamílias *Copia* e *Gypsy*, utilizando como treinamento sequências de DNA de angiospermas. De oito modelos, três se mostraram eficientes e conseguiram classificar satisfatoriamente as sequências, além de se mostrarem potencialmente eficiente na classificação de dados de espécies de angiospermas que não estavam no conjunto de dados usado para treino. Também foi realizado comparação da classificação dos três modelos mais eficientes com a predição feita pelo programa Blast, o qual possui um algoritmo baseado em alinhamento de sequências, como resultado, obteve-se excelentes métricas de classificação, todavia, considerando 80% de identidade e 80% de cobertura para que houvesse uma predição, este não conseguiu classificar 30% das sequências.

Palavras-chaves: Retrotransposons; Aprendizado de máquina; Classificação.

Lista de Ilustrações

Lista de tabelas

Sumário

1 Introdução

Por muito tempo, acreditou-se que o DNA era uma macromolécula estática, até que no final da década de 40, o “DNA saltitante” foi observado pela citogeneticista Barbara McClintock em seus experimentos com genes do milho. Assim, descobriu-se a existência de elementos capazes de se locomover dentro do próprio genoma, onde se inserem em um lócus diferente (MCCLINTOCK, 1950). Essa descoberta rendeu o Nobel de Fisiologia ou Medicina de 1983 à McClintock e a quebra do dogma do DNA estático à ciência.

Na atualidade, os “DNAs saltitantes” constatados pela Dra. McClintock são chamados de Elementos Transponíveis (TEs). Os TEs fazem parte do grupo de sequências repetitivas, esquematizado na Figura 1, estes são encontrados nos genomas de eucariotos e procariotos. Sabemos que os TEs compõem cerca de 50\% do genoma humano (RICHARD; KERREST; DUJON, 2008) e podendo constituir até 80\% do genoma das plantas, como é o caso do milho (RAVINDRAN, 2012), já em procariotos, os elementos transponíveis compõem a maioria dos genomas de bactérias, podendo modificar a expressão em frequências similares e até superiores às taxas de mutação naturais (KLECKNER, 1981).

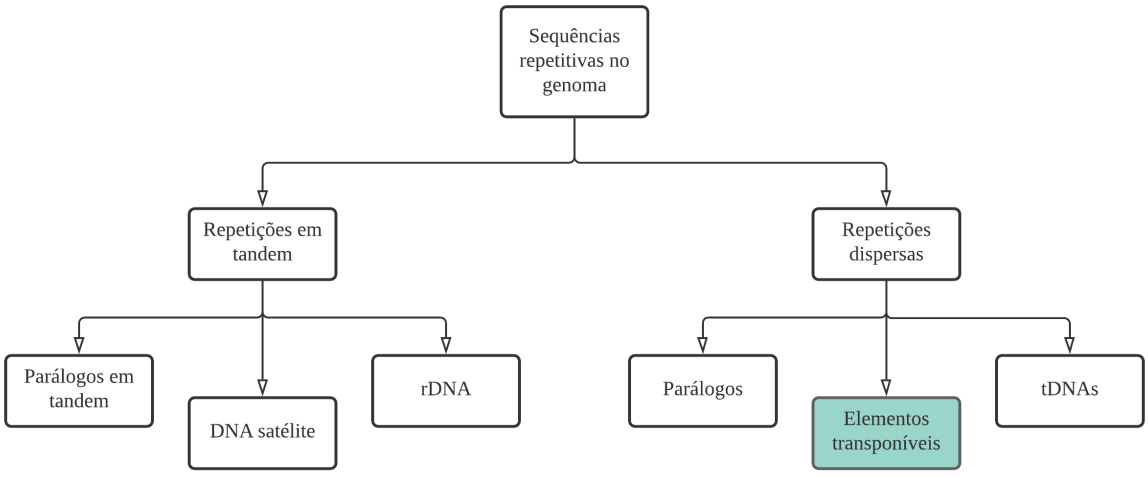

Figura 1 - Tipos de sequências repetitivas encontradas em genomas. Elaborado pelo autor.

A principal característica dos TEs é conseguir se locomover enzimaticamente de um lugar para outro dentro do genoma, podendo realizar uma cópia de si mesmo e inseri-la em um lócus distinto, mecanismo denominado copia e cola, ou simplesmente mudar sua sequência de lócus, mecanismo corta e cola (LEWIN, 2008). Todavia, apenas poucos TEs (<0,05%) continuam potencialmente móveis, os quais são chamados de TEs ativos, enquanto a maioria está inativa.

Durante anos, diversas pesquisas vêm sendo realizadas para elucidar as distintas funções que esses intrigantes fragmentos de DNA possuem. Dado a habilidade dos TEs em se inserirem em regiões aleatórias do DNA, sabe-se que podem influenciar na estrutura do genoma e na regulação da expressão de um gene (MITA; BOEKE, 2016), pois conseguem adentrar regiões de éxon e também em regiões de regulação, podendo causar efeitos deletérios, rearranjos genômicos ou até mesmo gerar um nova informação genética (MARTIENSSEN, 1998). Sendo assim, é possível que os elementos transponíveis sejam elementos da evolução, tornando-se ferramentas para a criação de um fenótipo que poderia levar ao aumento da sobrevivência de organismos individuais ou populações de organismos (KLECKNER, 1981).

Haja vista o avanço na biologia molecular e na bioinformática, as informações acerca dos TEs vem crescendo rapidamente e em grandes volumes, ademais, os TEs são extremamente diversos e possuem múltiplos mecanismos de replicação e mutabilidade, sendo assim, buscando unificar e organizar o sistema de nomenclatura e classificação dos TEs, Wicker e colaboradores (WICKER et al., 2007), em dezembro de 2007, propuseram o primeiro sistema de classificação taxonômica para elementos transponíveis de eucariotos, esquematizado na Figura 2, com base no mecanismo de transposição, semelhança entre as sequências de nucleotídeos e aminoácidos, além das relações estruturais. É nesse sistema de classificação que este projeto se baseia.

O sistema de Wicker é baseado no primeiro sistema de classificação de TEs proposto, criado por Finnegan, em 1989, no qual os TEs são divididos em classe I e classe II, sendo a primeira responsável por agrupamento dos elementos transponíveis que possuem um RNA intermediário, enquanto a classe II possui a ausência de RNA como intermediário (FINNEGAN, 1989). Todavia, essa classificação se tornou inviável porque associava a classe I ao mecanismo copia e cola e a classe II ao mecanismo corta e cola, porém atualmente sabe-se que existem TEs que utilizam o sistema corta e cola sem necessitar de um RNA intermediário (WICKER et al., 2007).

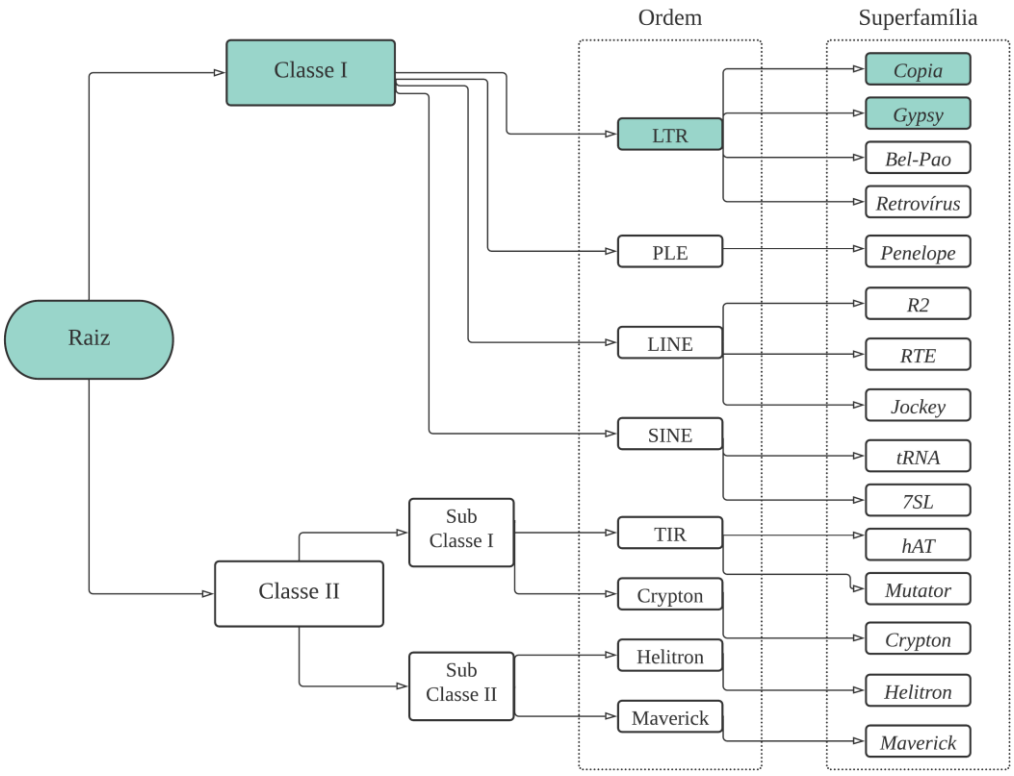

Figura 2 - Parte da classificação de TEs para eucariotos. Está destacado em azul a classificação de Copia e Gypsy. Adaptado de WICKER et al., 2007.

O sistema de classificação é hierarquizado em: classe, subclasse, ordem, superfamília e família; esses termos foram escolhidos de modo a espelhar a hierarquia já utilizada na filogenia de organismos. As classes mantêm a divisão antes feita por Finnegan, porém sem a associação do mecanismo de transferência, a qual é feita pelas subclasses, que dividem a classe II em subclasse I, agrupando elementos que utilizam o mecanismo copia e cola e subclasse II, mecanismo corta e cola. As ordens são divididas de acordo com a filogenia da transcriptase reversa, enquanto as superfamílias se diferenciam pelas suas proteínas e domínios não codificantes, a conservação da sequência no nível do DNA é pequena (WICKER et al., 2007).

Após superfamílias, os elementos são classificados em famílias (também chamadas de linhagens), é nesse agrupamento que *Copia* e *Gypsy* se encontram, todas as famílias do sistema de Wicker foram divididas de acordo com a conservação das sequências de nucleotídeos (WICKER et al., 2007). As superfamílias *Copia* e *Gypsy* (classe I, ordem LTR) são os TEs mais abundantes encontrados nos genomas de plantas (BENNETZEN; WANG, 2014) e são distinguidos entre si de acordo com a

ordem de seus domínios, como pode ser visto na Figura 3. Essas superfamílias são divididas em onze linhagens, *Athila, CRM, Del*, *Galadriel* e *Reina* pertencentes a *Gypsy* e *Ale, Angela, Bianca, Ivana, Maximus* e *Tar* pertencentes a *Copia*, Figura 4 (CRUZ et al., 2016).

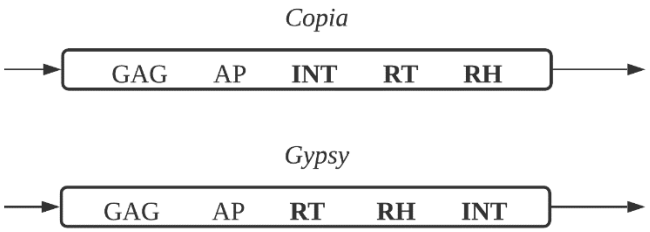

Figura 3 - Domínios de Copia e Gypsy. As setas indicam as regiões LTR, GAG = proteína do capsídeo, AP = Proteinase Aspártica, INT = Integrase, RT = Transcriptase Reversa, RH = RNase H. Adaptado de WICKER et al., 2007.

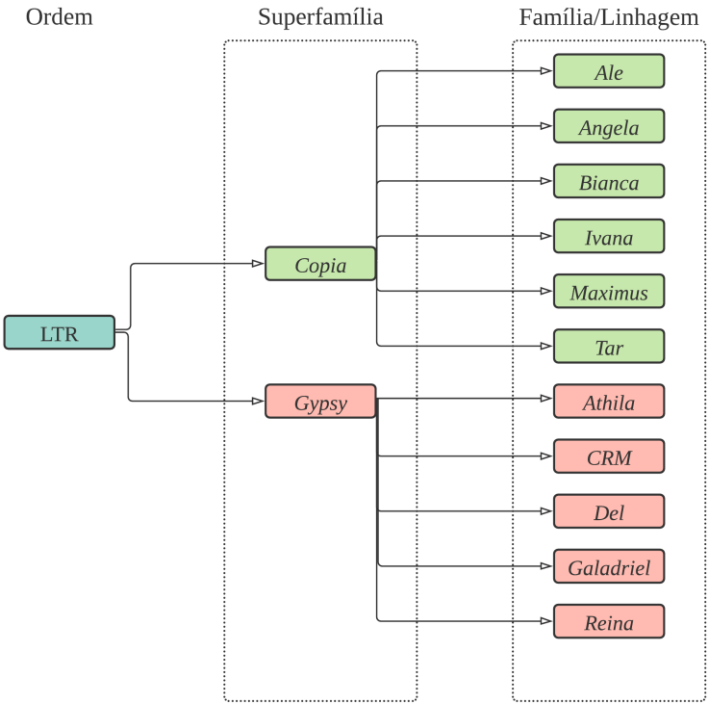

Figura 4 - Parte da classificação de TEs para eucariotos, retratando apenas a ordem LTR, duas de suas superfamílias e onze linhagens. Adaptado de CRUZ et al., 2016.

As superfamílias *Copia* e *Gypsy* não foram escolhidas ao acaso. Além de serem as mais abundantes em plantas, possuem o diferencial de estarem intimamente relacionadas com os retrovírus, o que pode ser constatado nas suas sequências e estruturas similares, além de apresentarem os sítios putativos (CHIU;

GRANDGENETT, 2000). Ademais, é possível que os retrovírus possam ter evoluído da superfamília *Gypsy* após a aquisição do gene do envelope (CRUZ et al., 2016).

A classificação de elementos tão distintos não é uma tarefa fácil, haja vista que são sequências repetitivas em abundância no genoma, por isso o uso da tecnologia se faz muito importante para o desenvolvimento de novos sistemas de identificação, assim, a utilização de aprendizado de máquina pode ser uma ferramenta importante nesse processo. Tendo isto em vista, este trabalho propõe classificar as onze linhagens das superfamílias dos retrotransposons *Copia* e *Gypsy* comparando diferentes algoritmos de Aprendizado de Máquina.

1.1 Problemática

Para fazer classificações de retrotransposons, no geral, as ferramentas mais utilizadas baseiam-se em alinhamento, como é o caso do Blast (ALTSCHUL et al., 1990; https://blast.ncbi.nlm.nih.gov/Blast.cgi), RepeatMasker (desenvolvido por A.F.A. Smith, não publicado, http://www.repeatmasker.org/) e LTR STRUC (MCCARTHY; MCDONALD, 2003), ambas são bastante aceitas pela comunidade científica. Também existem abordagens para essa classificação baseadas em aprendizado de máquina, como é o caso do trabalho de Nakano e colaboradores (NAKANO et al., 2017) e da ferramenta TEclass (ABRUSÁN et al., 2009), todavia, não classificam a nível de família para *Copia* e *Gypsy*.

No geral, para classificar uma nova sequência de retrotransposon e outros elementos transponíveis, é necessário conhecer a sequência nucleotídica deste e identificar seus domínios específicos, como exemplificados em *Copia* e *Gypsy* na Figura 3, estes domínios são encontrados por meio de alinhamento, identificar cada região da sequência como um domínio gasta tempo e requer uma análise minuciosa do resultado pelo pesquisador.

Sendo assim, a pergunta desse trabalho é se é possível conseguir classificar linhagens de retrotransposons *Copia* e *Gypsy* utilizando apenas algoritmos de aprendizado de máquina e sequências de nucleotídeos de angiospermas, sem se importar com a diferenciação dos domínios de cada sequência, de forma a automatizar o processo e encontrar padrões não tão visíveis como os encontrados pelo alinhamento. Classificar um novo TE como ordem LTR pode ser feito apenas encontrando a presença das sequências de LTRs, todavia a classificação em uma das

superfamílias e famílias pode não ser tão simples devido à falta de informação clara na sequência (WICKER et al., 2007).

1.2 Objetivo

1.2.1 Objetivo Geral

O objetivo geral desta pesquisa foi a criação de um método eficiente e automático que pudesse ser aplicado na classificação taxonômica de sequências de retrotransposons de superfamílias *Copia* e *Gypsy*, distinguindo-os pelas suas linhagens, utilizando diferentes algoritmos de aprendizado de máquina, além de comparar os resultados obtidos pelo Blast para os mesmos conjuntos de dados. Ao final do projeto, além de avaliarmos se o objetivo geral foi atingido comparando métricas de avaliação comuns no ramo do aprendizado de máquina, também possuímos como objetivo analisar a classificação de sequências de retrotransposons de angiospermas que não estavam no nosso grupo utilizado para a realização dos conjuntos de dados, de forma a relatar um potencial em nossos modelos para a aplicação em outras espécies.

1.2.2 Etapas de Desenvolvimento

Considerando o objetivo geral apresentado, é possível enumerar as etapas desenvolvidas para alcançá-lo:

1. Análise dos dados iniciais e construção dos conjuntos de dados: esta etapa envolveu as sequências fastas obtidas do artigo de CRUZ et al (CRUZ et al., 2016) para a construção de dois conjuntos de dados, ambos em formato csv, contendo os atributos e classes de cada sequência.
2. Implementação dos algoritmos de Aprendizado de Máquina: utilizou-se três algoritmos de aprendizado de máquina: kNN, árvore de decisão e rede neural, ambos serão detalhados no capítulo de Fundamentação Teórica. Nessa etapa, estes foram usados em *scripts* na linguagem Python 3.
3. Análise dos resultados: Obtenção das métricas resultantes dos algoritmos executados e aplicação de cálculos estatísticos para avaliar a confiança estatística dos resultados.

4. Comparação dos resultados: Nessa etapa foi escrito um script para criar diversos bancos de dados no software Blast com os dados obtidos para treinamento dos algoritmos de aprendizado de máquina, a fim de analisar a classificação feita por estes e comparar com nossos resultados.
5. Utilização dos modelos de Rede Neural para classificar sequências nucleotídicas de Cana de açúcar.

2 Fundamentação teórica

2.1 Aprendizado de Máquina

Com o advento do sequenciamento de nova geração e desenvolvimento de novas técnicas moleculares, obter uma sequência genômica se tornou mais rápido e mais barato em comparação com décadas anteriores, consequentemente, existe uma grande quantidade de dados biológicos disponíveis. Para lidar com toda essa informação, surge a necessidade de ferramentas automáticas e que possam aprender com os dados à medida que os analisa, por isso a técnica de aprendizado de máquina vem crescendo e se destacando.

O aprendizado de máquina faz parte do que chamamos de inteligência artificial, uma área de estudo que objetiva explicar e simular o comportamento inteligente em termos de processos computacionais (SCHALKOFF, 1990). O nome aprendizado de máquina não é em vão, de fato, o algoritmo precisa ter a capacidade de aprender, e assim criar modelos matemáticos para realizar a sua principal tarefa que é fazer inferências. O modelo pode ser preditivo, descritivo ou ambos (ALPAYDIN, 2010).

O aprendizado de máquina se baseia em padrões, tanto em detectá-los nos dados como extrapolá-los para fazer deduções, os quais auxiliam a entender e reconhecer o processo ou um objeto como um todo (RUSSELL; NORVIG, 1995). Para Arthur Samuel, "Programar computadores para aprender com a experiência deve eliminar a necessidade de grande parte do esforço de programação detalhado" (SAMUEL, 1959), ou seja, utilizar algoritmos de aprendizado de máquina podem diminuir consideravelmente o tempo e o esforço para se chegar em conclusões eficientes.

Existem diferentes formas de se abordar um problema de aprendizado de máquina, o que normalmente é determinado pelos dados e pelo objetivo. Nesse trabalho foi utilizado algoritmos de aprendizado de máquina supervisionado para classificação, esta e outras abordagens podem ser visualizadas na Figura 5.

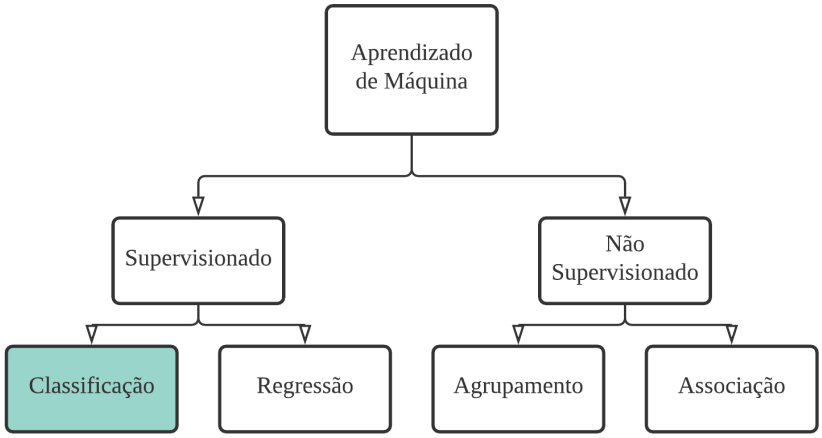

Figura 5 - Algumas abordagens de aprendizado de máquina. Elaborado pelo autor.

2.1.1 Classificação

Para entender a classificação é importante destacar alguns conceitos, como: instância é o exemplo utilizado em todo o conjunto de dados, possui atributos e classes; atributo é uma característica da instância, a qual pode ser nominal ou contínua; enquanto classe, também chamada de rótulo é o que se deseja concluir ao fazer previsões utilizando o modelo de aprendizado supervisionado, também pode ser nominal ou contínua (REZENDE, 2003).

É necessário dividir o conjunto de dados, ou seja, as instâncias, em dois conjuntos: o conjunto de treinamento e o de teste. O primeiro é disponibilizado para o algoritmo extrair e analisar padrões, esse conjunto tem um impacto significativo no sucesso do modelo, sendo assim, precisa conter dados assertivos que representem a distribuição de exemplos sobre os quais o desempenho do sistema será mensurado (MITCHELL, 1997). Já o conjunto de teste é crucial para afirmar que o modelo de fato está inferindo os resultados corretamente, geralmente representa cerca de 20% do conjunto de dados total, mas não há regra para essa divisão.

A classificação se baseia na previsão de rótulos, resumidamente, o que ocorre é a definição de um modelo sobre todo o espaço de entrada e parâmetros, sendo assim, uma vez que tenhamos um padrão encontrado que se ajusta aos dados do treinamento, e se o dado não rotulado for semelhante ao treinado, então se pode fazer previsões corretas para novas instâncias (ALPAYDIN, 2010).

2.2 Algoritmos de aprendizagem

2.2.1 Algoritmo dos vizinhos mais próximos (kNN)

É um algoritmo de classificação não-linear e não paramétrico que possui o seguinte funcionamento: dado n vetores de treinamento, o algoritmo identifica os k vizinhos mais próximos destes e suas classes; k é um número positivo e inteiro, geralmente ímpar e não múltiplo do número de classes totais. Por fim, é atribuído, como rótulo a uma instância desconhecida, a classe predominante na vizinhança (THEODORIDIS; KOUTROUMBAS, 2009), este esquema pode ser visualizado na Figura 6. É importante destacar que o processo da escolha do valor de k não é trivial, pois um k alto ocasiona um modelo simples, um alto viés e uma baixa variância, enquanto um k baixo leva a um estimador com variância alta, mas viés baixo (IZBICKI; SANTOS, 2020).

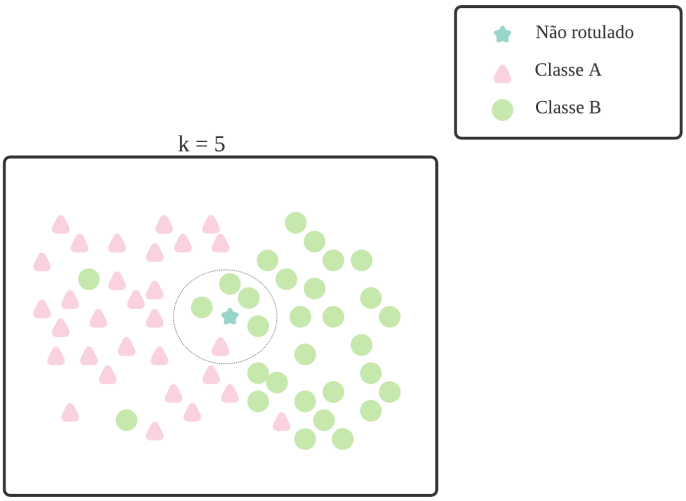

Figura 6 - Um possível plano com 2 classes e uma instância não rotulada.

A hipótese base do kNN realiza a associação de uma classe desconhecida, possivelmente pode ser da classe predominante na vizinhança. Para calcular quais

serão os vizinhos próximos, é preciso utilizar a métrica de distância, a qual pode ser calculada de diferentes fórmulas é mais comum que se utilize a distância generalizada Minkowski (equação 2.1), a qual varia de acordo com o valor de p. Para p=2, distância euclidiana. Para p=1, distância Manhatthan. Para p=0, distância Hamming.

$$\sqrt[p]{(\sum_{i-1}^{n} |x_i - y_i|^p} \tag{2.1}$$

2.2.2 Árvore de decisão

Na classificação, o algoritmo de árvore de decisão destaca-se em várias aplicações por sua simplicidade e interpretabilidade. Se baseia em classificar as instâncias utilizando nós, os quais são interligados por ramos, cada nó simboliza um teste sobre algum atributo. O primeiro nó de decisão da árvore é chamado de raiz, a partir dele o algoritmo divide os dados de acordo com os dados que correspondem à possibilidade do ramo, os últimos nós da árvore são chamados de nós folhas e não possuem ramos saindo deles e assim a instância é classificada (MITCHELL, 1997). Nesse processo, em cada nível da árvore, um teste é aplicado em algum atributo, para decidir qual o próximo nó a seguir, este esquema pode ser visualizado na Figura 7.

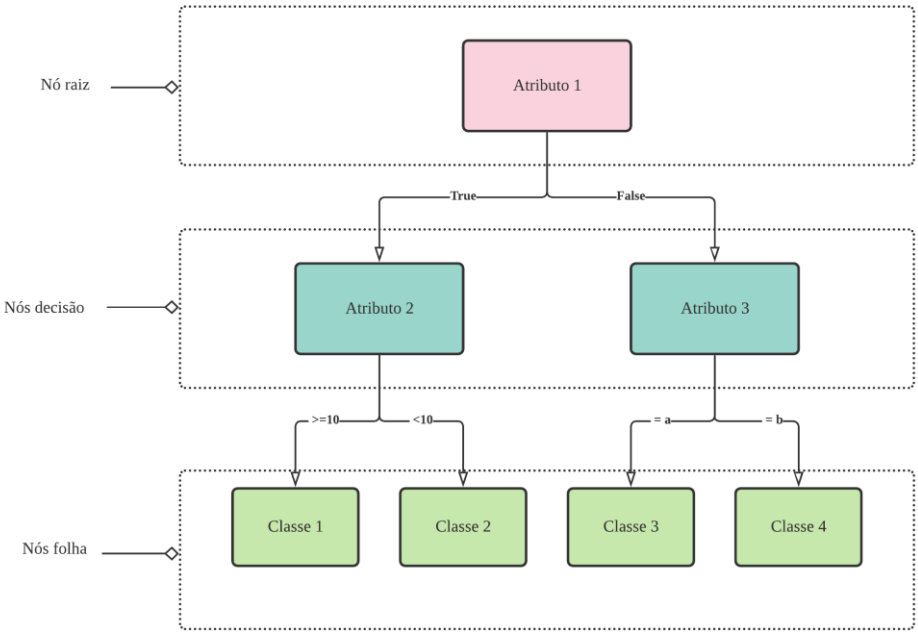

Figura 7 - Exemplo genérico de uma possível árvore de decisão. Elaborado pelo autor.

Na construção de uma árvore de decisão, é necessário escolher critérios para definir qual será a melhor disposição dos nós e, consequentemente, partição dos dados. Existem diversos critérios que podem ser utilizados nesse particionamento, como: entropia e gini. Nesse trabalho, utilizamos a entropia que está intimamente relacionada com a probabilidade, fato demonstrado pelo cientista Boltzmann, que afirma que a entropia varia de acordo com o número de modos que o sistema pode ser montado, a partir dessa relação surgiu a equação da entropia aplicada às árvores de decisão, como mostrado na Equação 2.2, onde p é a probabilidade de cada classe J acontecer e c é o número de classes (DRECHSLER, 1968).

$$-\sum_{j=1}^{c} p_j . log_2(p_j) \tag{2.2}$$

A Equação 2.2 é utilizada para calcular a entropia das diferentes disposições possíveis dos atributos, sabendo-se que se aumenta a entropia, aumenta também a desordem e consequentemente diminui a informação. Assim, ao avaliar qual a melhor partição de atributos e montar a árvore, o algoritmo escolhe aquela com a menor entropia, pois consequentemente esta terá a menor desordem e suas chances de erros serão minimizadas.

As árvores aprendidas também podem ser representadas novamente como conjuntos de regras se-então para melhorar a legibilidade humana. Seu viés indutivo é uma preferência por árvores pequenas em vez de árvores grandes"(MITCHELL, 1997).

2.2.3 Rede Neural Artificial (RNA)

Neste trabalho, também utilizamos Redes Neurais Artificiais (RNA), as quais possuem um sistema lógico-matemático baseado na funcionalidade dos neurônios do cérebro humano. O comportamento das conexões sinápticas entre os neurônios do cérebro é simulado por meio de pesos na RNA, a qual é normalmente utilizada para classificação e regressão (GURNEY, 1997).

As Redes Neurais Artificiais são organizadas em camadas, a primeira camada é por onde os dados são inseridos, esses valores que adentram a rede são multiplicados por pesos, que podem ser números inicialmente aleatório e podem ser positivos ou negativos; a última camada é a de saída, que contém os valores finais após terem passado por todo o processo da rede neural. As camadas que não são de entrada e saída, são as camadas ocultas, uma Rede Neural pode ter uma ou mais destas. Existem arquiteturas que fogem a esse padrão, mas não serão utilizadas nesse trabalho.

A informação ao longo dessa Rede Neural inicia adentrando a rede pelos neurônios da camada de entrada, estes são multiplicados pelos pesos sinápticos, cada neurônio de entrada é conectado a todos os neurônios da camada oculta, assim, cada conexão possui um peso sináptico. Esses resultados são somados, o resultado dessa soma entre em uma função, a qual chamamos de função de ativação. Após a função, os resultados saem pelo neurônio(s) da camada de saída (SILVA; SPATTI; FLAUZINO, 2010), esquematizado na Figura 8.

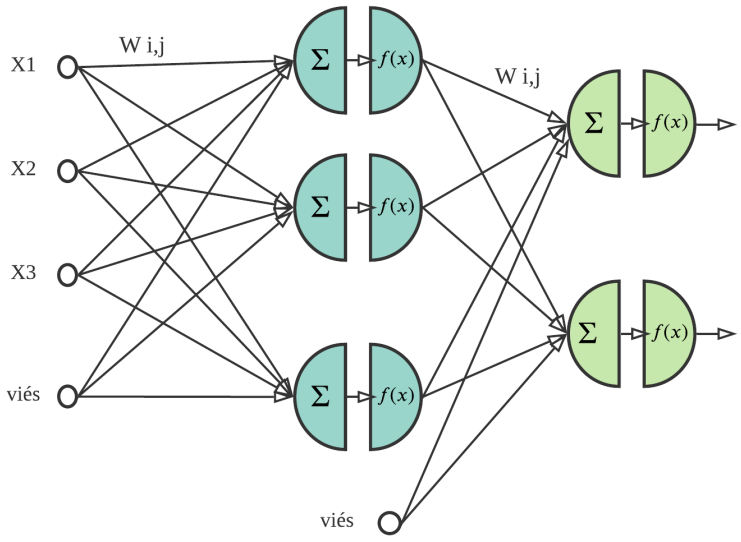

Figura 8 - Esquema genérico do fluxo de informações em uma possível Rede Neural Artificial, onde Wi,j são os pesos. Elaborado pelo autor.

2.3 BLAST

O BLAST é um algoritmo de busca que utiliza método de palavras mesclado com programação dinâmica. Ao fornecer uma ou mais sequências de nucleotídeos para o BLAST, este diminui as sequências em subsequências de tamanho w, a qual

chama de palavras. Essas palavras são alinhadas ao banco de dados, ao encontrar um match, o algoritmo estende o alinhamento para a esquerda e para a direita da palavra e utiliza a lógica de Smith-Waterman modificada para calcular o melhor alinhamento e pontuações.

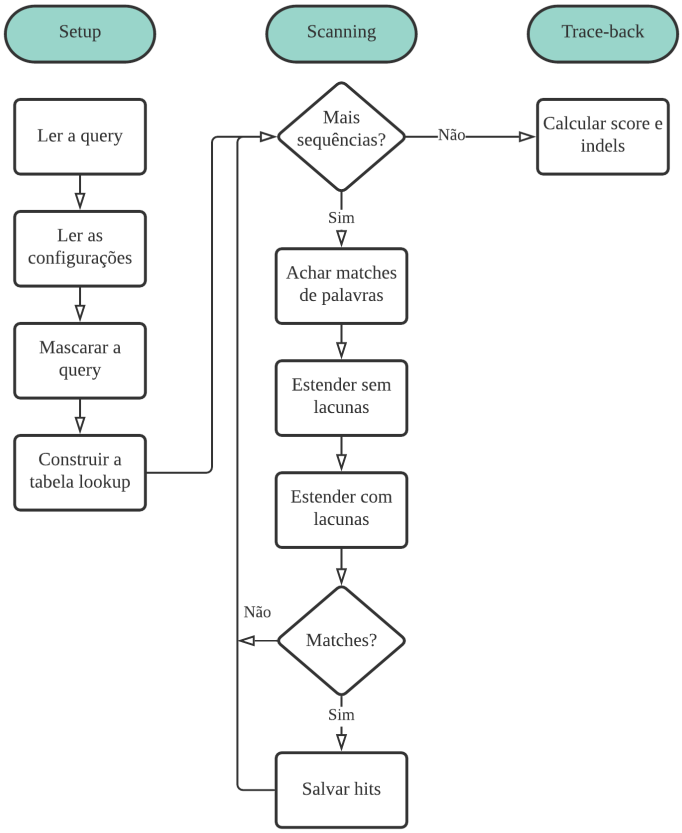

Figura 9 - Etapas do funcionamento interno do BLAST. Elaborado pelo autor.

3 Metodologia

A linguagem de programação utilizada em todo o projeto foi o Python versão 3 e sua biblioteca Biopython versão 1.73.

3.1 Obtenção das sequências de retrotransposons

As 28.622 sequências de DNA das onze linhagens de *Gypsy* e *Copia* foram obtidas do material suplementar do artigo "Virus-like attachment sites as structural landmarks of plants retrotransposons" (CRUZ et al., 2016). Cada linhagem estava em

um arquivo fasta, resultando em onze arquivos distintos, os quais foram unidos para gerar um arquivo multifasta geral. Um arquivo em formato fasta possui um padrão: a primeira linha que indica uma sequência sempre inicia com o símbolo “>” e é seguido do nome da sequência, que pode conter um id, a espécie, entre outros; logo abaixo dessa linha inicial encontra-se a sequência de fato.

3.2 Criação dos conjuntos de dados

Foram construídos dois conjuntos de dados específicos para aprendizado de máquina, com o objetivo de extrair atributos das sequências. O primeiro conjunto de dados foi constituído por k-mers, os quais dizem respeito a uma contagem de substrings (MARÇAIS; KINGSFORD, 2011), ou seja, a uma contagem de todas as possíveis combinações que podem ocorrer de comprimento igual a k. Neste projeto, utilizamos k igual a 2, 3 e 4, totalizando 336 atributos. Dessa forma, um arquivo final em formato csv foi construído, contendo os identificadores das onze linhagens e os k-mers de cada sequência.

$$\mathrm{DAC}(u, lag) = \sum_{i=1}^{L-lag-1} (P_u(\mathrm{R}_i\mathrm{R}_{i+1}) - \overline{P}_u)(P_u(\mathrm{R}_{i+lag}\mathrm{R}_{i+lag+1}) - \overline{P}_u) / (L - lag - 1) \quad (3.1)$$

O segundo conjunto de dados foi criado com base no mesmo arquivo fasta com as 28.622 sequências, todavia, aplicando-se o pacote Pse-in-One 2.0 (LIU et al., 2017), o qual correlaciona informações provenientes das sequências de nucleotídeos ou aminoácidos em informações quantitativas. As ferramentas do pacote são divididas de acordo com o tipo de sequência: DNA, RNA ou proteína; nesse projeto, utilizou-se as ferramentas atribuídas às sequências de DNA, mais especificamente, utilizamos o método Dinucleotide-based autocovariance (DAC), que pode ser visualizada na equação 3.1, o qual mede a correlação do mesmo índice físico-químico entre dois dinucleotídeos separados por uma distância lag ao longo da sequência, definiu-se lag como 1 e utilizou-se as 148 características físico-químicas disponíveis para DNA: entalpia, inclinação, entropia, conteúdo GC, flexibilidade, entre outras que podem ser conferidas em <https://github.com/banshanren/Pse-in-One-2.0/blob/master/docs/Pse-in-One\%202.0_description.pdf>

3.3 Algoritmos de Aprendizado de Máquina

Em todos os algoritmos foi realizado padronização inicial e validação aplicando a técnica de *Nested Cross-Validation*, esquematizada na Figura 10, a qual divide os dados diversas vezes em conjuntos de treinamento e teste para estimar a precisão da generalização de um classificador a fim de diminuir sobre ajuste (*overfitting*) e incorporar de maneira eficaz o ajuste de parâmetros para treinar um modelo de previsão ideal (PARVANDEH et al., 2020), utilizou-se dez partições externas e dez partições internas, na qual as classes KFold e GridSearchCV da biblioteca *scikit-learn* (https://scikit-learn.org/stable/) foram utilizadas para repartir os dados em treino e teste e otimizar os parâmetros por uma pesquisa em grade com validação cruzada (*cross-validation*) na partição interna e validá-la no teste externo. Vale destacar que a técnica de *Nested Cross-Validation* necessita de muito poder computacional. A biblioteca scikit-learn foi utilizada na construção de todos os modelos utilizados nesse projeto: algoritmo dos vizinhos próximo, árvore de decisão e rede neural.

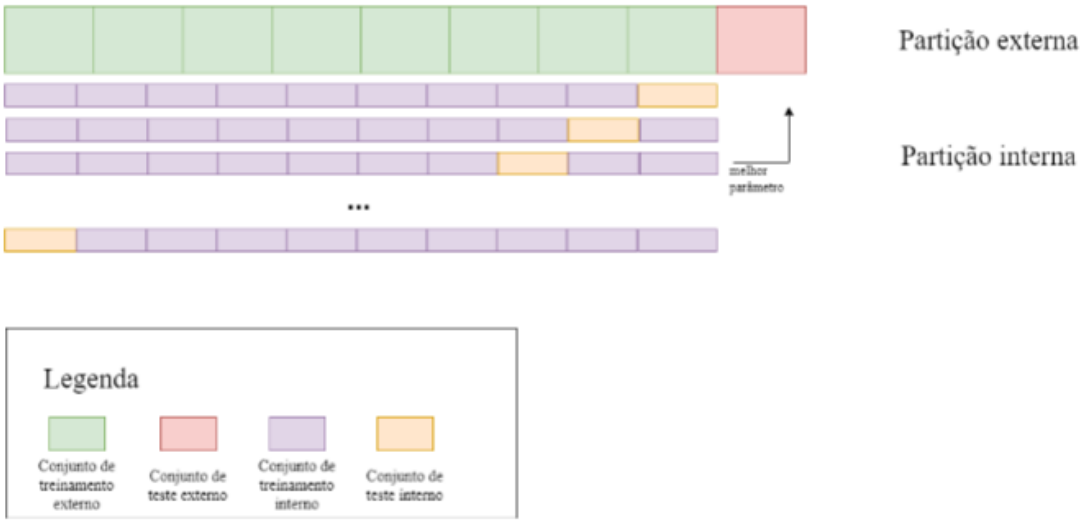

Figura 10 - Esquema representativo do *Nested Cross-Validation*. Elaborado pelo autor.

3.3.1 Algoritmo dos vizinhos mais próximos (kNN)

Para a aplicação do algoritmo kNN, dividiu-se os dados padronizados em dados com redução de dimensionalidade PCA e sem PCA. Logo, foram quatro algoritmos de kNN empregados: kNN com PCA utilizando método DAC, kNN com PCA

utilizando k-mers, kNN sem PCA utilizando método DAC e kNN sem PCA utilizando k-mers. Para o parâmetro de distância no algoritmo, utilizou-se a distância Minkowski (equação 3.2), a qual é a métrica generalizada de distância de um espaço bi-dimensional, com p=2 (equação 3.3) representando a clássica distância euclidiana, assim, calculando a menor distância entre dois pontos em um ambiente plano (GERR, 1995).

$$D(x, x') = \sqrt[p]{\sum_{d} |x_d - x'_d| p} \tag{3.2}$$

$$d(x, y) = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2 + ... + (x_n - y_n)^2} \tag{3.3}$$

No *Nested Cross-Validation*, o parâmetro utilizado foi o k, o qual determina o número de vizinhos que serão analisados para categorizar o novo ponto não rotulado, os k disponibilizados para o algoritmo foram 1, 3, 5, 7, 9, 11, 13, 15, 17.

3.3.2 Árvore de Decisão

No algoritmo de árvore de decisão não houve aplicação da técnica de redução de dimensionalidade PCA. Assim, foi desenvolvido dois algoritmos: árvore de decisão utilizando método DAC e árvore de decisão utilizando k-mers. Nesse projeto, utilizou-se a entropia (S) como critério de particionamento dos atributos, mesmo essa gastando maior poder computacional por utilizar log em sua fórmula (equação 3.4) em comparação com outro critério muito utilizado: gini (equação 3.5). Escolheu-se a entropia porque está intimamente relacionada com a probabilidade e ser amplamente utilizada na academia. A entropia varia de acordo com o número de modos que o sistema pode ser montado, assim, ao avaliar qual a melhor partição de atributos, o algoritmo escolhe aquela com a menor entropia, pois consequentemente esta terá a menor desordem e suas chances de erros serão mínimas (DRECHSLER, 1968).

$$S = \sum_{i=1}^{c} p_i . log_2 . p_i \tag{3.4}$$

$$Gini = 1 - \sum_{i=1}^{c} (p_i)^2 \tag{3.5}$$

Na *Nested Cross-Validation*, utilizou-se a profundidade da árvore como parâmetro para ser selecionado, foi fornecida uma lista de 3 a 12 para ser selecionado o melhor.

3.3.3 Redes Neurais Artificiais

No algoritmo de redes neurais, não houve aplicação da técnica de redução de dimensionalidade PCA. Para esse algoritmo, utilizou-se 174 neurônios em uma única camada oculta, além de ser aplicado camada precoce ativa, ou seja, o algoritmo para quando a diferença entre o erro da iteração presente com a iteração anterior não for menor que 10^{-8}, o otimizador utilizado foi Adam, o qual é computacionalmente eficiente, tem poucos requisitos de memória e é de fácil implementação em conjuntos de dados pequenos, apropriado para ser utilizado em objetivos não estacionários (KINGMAN; BA, 2015) adjunto da função Relu como função de ativação. Os parâmetros utilizados na *Nested Cross-Validation* foram o tamanho do batch entre 1, 50, 100.

3.4 Métricas de avaliação

Para se realizar a mensuração do desempenho, utilizou-se medidas de avaliação: acurácia, precisão, revocação (também chamada de sensibilidade) e F1-score. A acurácia (equação 3.6) indica a porcentagem de acertos do modelo. As siglas TP, TN, FP, FN equivalem, respectivamente, às classes positivas previstas corretamente, classes negativas previstas corretamente, classes falso-positivas (descritas como positivas, porém são negativas) e classes falso-negativo (descritas como negativa, porém são positivas).

$$Acuracia = \frac{TP + TN}{TP + TN + FP + FN} \tag{3.6}$$

A precisão (equação 3.7) expressa a possibilidade de uma classe positiva ser um TP entre todos os resultados classificados pelo modelo como positivos para uma determinada classe, já a revocação (equação 3.8) diz respeito à possibilidade de classificação de um resultado da classe positiva ter sido classificado corretamente entre todos os que de fato eram positivos, os conceitos podem ser facilmente confundidos, por isso se dá a importância de expressá-los matematicamente para melhor compreensão. Tanto a precisão, quanto a revocação são medidas de relevância (FAWCETT, 2005). A métrica F1-score (equação 3.9) é uma média harmônica entre a precisão e a revocação de forma a chegar em uma valor que represente ambos.

$$Precisao = \frac{TP}{TP + FP} \tag{3.7}$$

$$Revocacao = \frac{TP}{TP + FN} \tag{3.8}$$

$$F1 - score = \frac{2x(PrecisaoxRevocacao)}{Precisao + Revocacao} \tag{3.9}$$

Ambas as métricas foram calculadas utilizando a média macro entre as classes, ou seja, cada métrica foi calculada para cada classe e posteriormente uma média aritmética foi realizada; esse processo foi refeito para cada iteração no *Nested Cross-Validation*, o qual possuía 10 partições, sendo assim, foram realizadas 10 iterações e, consequentemente, 10 macro médias de cada métrica de avaliação. Para obter os resultados, foi realizado o cálculo da média aritmética das 10 macro médias existentes.

3.5 BLAST

Para comparar as métricas obtidas nos algoritmos de aprendizado de máquina com o algoritmo do programa Blast, foi necessário criar bancos de dados Blast idênticos aos conjuntos de dados treinamento, na tentativa de rotular as sequências utilizadas em nosso conjunto teste utilizando o alinhamento do Blast,

sendo assim, foi criado um script em Python 3 que realizava um *cross-validation* com as sequências: as sequências separadas para treinamento eram usadas para a criação do banco de dados e as de teste eram utilizadas para a validação contra o banco de dados em cada loop, foram realizados 10 loops, assim como nos algoritmos de aprendizado de máquina.

As sequências testes comparadas ao banco de dados criado no Blast eram classificadas apenas se o algoritmo Blast encontrasse outra sequência que possuísse pelo menos 80% de identidade e 80% de cobertura que é o mínimo para classificar sequências de TEs pertences à mesma família utilizando alinhamento (WICKER et al., 2007).

4 Resultados e Discussão

4.1 Extração de informações iniciais das sequências

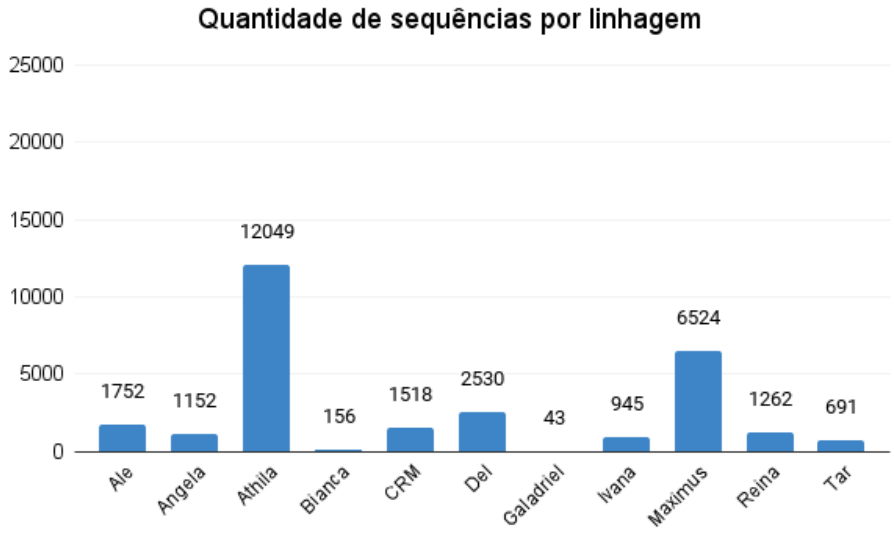

Figura 11 - Quantidade de sequências por linhagem no arquivo fasta inicial. Elaborado pelo autor.

No arquivo fasta inicial, algumas sequências se encontravam em maior abundância, como é o caso de Athila e Maximus, visto na Figura 11, enquanto as famílias Bianca e Galadriel estão pouco representadas, esse último fato ocorre porque essas famílias são pouco encontradas nos genomas das angiospermas analisados, os quais são de cinco eudicotiledôneas (*Arabidopsis thaliana, Medicago truncatula, Populus*

trichocarpa, Vitis vinifera e *Glycine max*) e cinco monocotiledôneas (*Brachypodium dystachyon, Oryza sativa, Setaria bicolorica, Sorghum* e *Zea mays*).

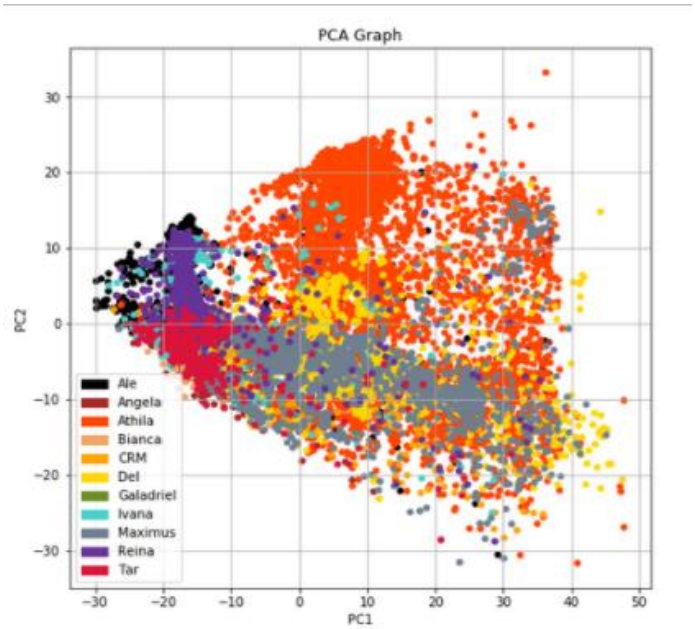

Figura 12 - Distribuição do conjunto de dados k-mers utilizando PCA. Elaborado pelo autor.

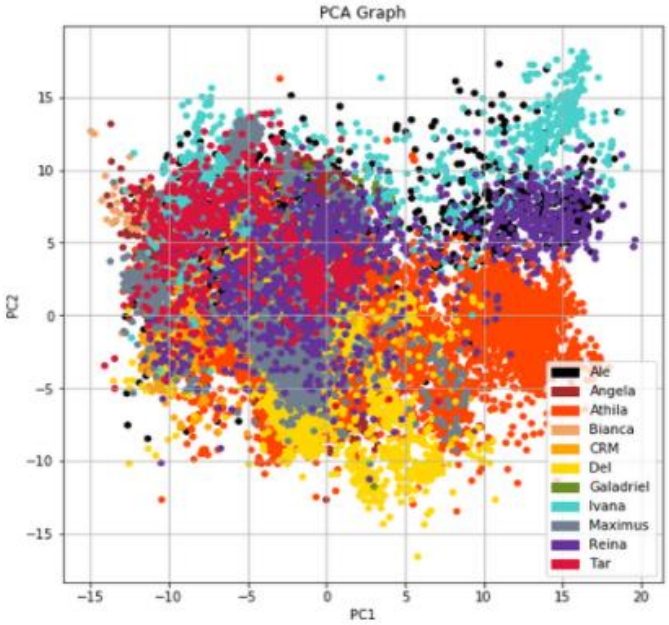

Figura 13 - Distribuição do conjunto de dados DAC utilizando PCA. Elaborado pelo autor.

Ademais, as distribuições dos conjuntos de dados realizados a partir do arquivo fasta inicial podem ser visualizadas utilizando a técnica de PCA em gráficos, foi criado um gráfico bidimensional com os vértices sendo PC1 e PC2 para k-mers e

para o conjunto de dados realizado com pse-in-one (DAC), na Figura 12 e Figura 13, respectivamente.

4.2 Métricas de avaliação dos algoritmos de aprendizado de máquina

Tabela 1 - Métricas de avaliação obtidas em cada algoritmo. Elaborado pelo autor.

Algoritmo	Acurácia	Precisão	Revocação	F1-score
kNN sem PCA - k-mer	0.8920	0.7258	0.8898	0.7949
kNN sem PCA - DAC	0.9385	0.9433	0.8725	0.8986
kNN com PCA - k-mer	0.7532	0.7348	0.5420	0.5986
kNN com PCA - DAC	0.6310	0.7208	0.3450	0.4087
Árvore de Decisão - k-mer	0.8665	0.8492	0.7688	0.7988
Árvore de Decisão - DAC	0.8228	0.8732	0.6807	0.7294
Rede Neural - k-mer	0.9573	0.9660	0.9275	0.9439
Rede Neural - DAC	0.9402	0.9136	0.8727	0.8845

As métricas de avaliação de ambos os algoritmos foram comparadas entre si, os valores obtidos como resultados podem ser observados na Tabela 1. É notório que kNN com PCA - DAC obteve o pior resultado entre os demais, haja vista que a revocação assumiu o valor de 0,3450, o que pode representar um alto número de falsos negativos nesse modelo, em contrapartida, nenhum outro método obteve valores de métricas de avaliação menor que 0,50.

Ao se analisar apenas kNN e comparar os resultados obtidos quando se aplicou a técnica PCA e quando não aplicou, constata-se que os dados sem PCA obtiveram melhores resultados, tanto utilizando k-mers quanto utilizando o método DAC, essa questão pode ter acontecido porque mesmo PC1 e PC2 representando a maior variação, os outros dados possuíam informações significativas para a classificação, de forma que quando se considerou os dados por completo, a classificação se tornou mais eficiente.

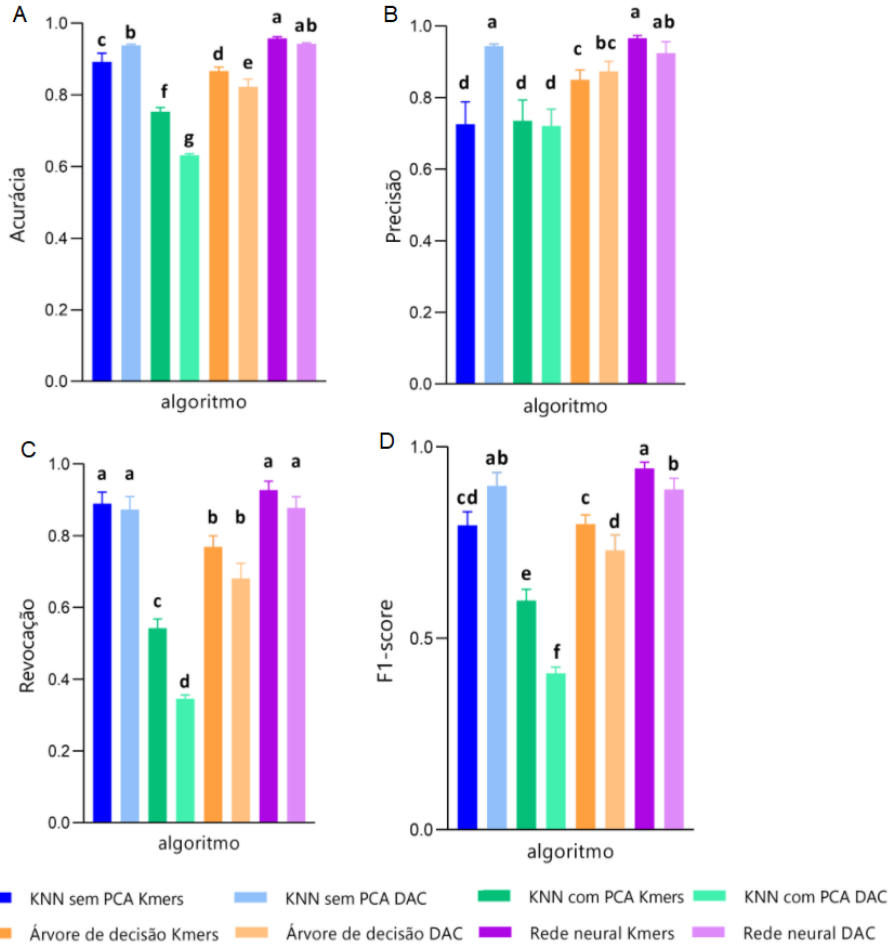

Figura 14 - Gráficos obtidos com as técnicas Anova e teste tukey. Elaborado pelo autor.

Para analisar melhor os resultados, realizou-se o teste estatístico Anova conjuntamente com o teste Tukey para múltipla comparação das médias levando em conta os desvios padrões, as comparações podem ser observadas na Figura 14; estes gráficos servem para uma comparação eficiente e estatística dos resultados, cada barra contém a média e o desvio padrão referente a mesma métrica de avaliação nos diferentes algoritmos.

As letras acima de cada barra servem como resultado de comparação desta com as demais médias; o teste estatístico associa a letra “a” à maior média e elenca em ordem alfabética as demais, até a menor média que possuirá a letra mais distante de “a” no alfabeto. Quando uma barra possui mais de uma letra indica que as barras referentes a essas letras não conseguem se diferenciar estatisticamente, ou seja, o p valor foi maior que 0,01 (por exemplo “ab” significa que esta não difere em relação à “a” nem à “b”); consequentemente, barras com letras diferentes são distintas estatisticamente, pois os p valores são menores que 0,01.

Portanto, foi concluído nessa primeira parte que três algoritmos de aprendizado de máquina mais se destacaram: kNN sem PCA DAC, Rede Neural k-mers e Rede Neural DAC, não é possível concluir apenas um pois vale notar que ao comparar o conjunto de dados de k-mers com o de características físico-químicas (utilizadas no DAC), o primeiro é de menor complexidade. Ademais, ao compararmos o algoritmo de Rede Neural com o kNN, apesar do primeiro ter obtido ligeiramente maiores métricas, o kNN traz como vantagem a rapidez e a menor exigência de processamento.

4.3 Comparação dos resultados com o Blast

De um total de 28.622 sequências, o algoritmo de alinhamento do Blast conseguiu classificar apenas 17.435 sequências, devido a regra de 80-80, ou seja, necessitar que a sequência possuísse pelo menos 80% de identidade e 80% de cobertura para predizer. As métricas expostas na Tabela 2 são comparações entre as mesmas sequências classificadas pelos diferentes algoritmos.

É notório que o algoritmo do Blast possui as métricas mais altas, mas as diferenças em comparação com os algoritmos de aprendizado de máquina são mínimas que se tornam irrelevantes, sendo assim, é possível concluir que todos conseguiriam classificar muito bem as 17.435 sequências, entretanto o algoritmo do Blast, por não conseguir classificar 11.187, mostrou-se menos eficiente.

Tabela 2 - Métricas de avaliação obtidas pelas mesmas 17.435 sequências.

Algoritmo	Acurácia	Precisão	Revocação	F1-score
BLAST	0.9795	0.9707	0.9751	0.9729
kNN sem PCA - DAC	0.9656	0.9457	0.9055	0.9232
Rede Neural - k-mer	0.9748	0.9580	0.9336	0.9444
Rede Neural - DAC	0.9691	0.9434	0.9194	0.9308

4.4 Utilizando dos modelos de rede neural para classificar outras sequências

Visando utilizar os modelos que obtiveram bons resultados para classificar outros dados, criamos um conjunto de dados a partir de um arquivo fasta fornecidos

pela Prof. Drª. Marie-Anne van Sluys, atualmente professora na Universidade de São Paulo (USP), o qual continha 58 sequências nucleotídicas de elementos transponíveis de cana de açúcar, angiosperma esta que não foi usada como treinamento para o modelo. Entre essas sequências, algumas eram rotuladas como famílias que não estavam entre as 11 linhagens que o modelo consegue classificar, estas são: MUTATOR, HAT, CACTA e TAT.

Apesar do conjunto de dados ser considerado muito pequeno, os resultados se mostram bastante otimistas, o que pode ser visto na Tabela 3 e 4, contendo os acertos de cada algoritmo, porém ignorando as sequências que eram rotuladas como linhagens que o modelo não classificava por não conseguir se realizar comparação. Ambos os modelos demonstraram um ótimo desempenho, com uma diferença de acertos apenas na família Reina, em que o modelo construído com Rede Neural - DAC errou uma sequência, classificando-a como Del, ao invés de Reina.

Tabela 3 - Métricas de avaliação obtidas pelas 17.435 sequências com Rede Neural - k-mer.

Algoritmo	Linhagem	Total	Número de Acertos
Rede Neural - k-mer	DEL	6	6
Rede Neural - k-mer	REINA	6	6
Rede Neural - k-mer	ALE	6	6
Rede Neural - k-mer	TAR	4	2
Rede Neural - k-mer	ANGELA	3	2
Rede Neural - k-mer	IVANA	6	6
Rede Neural - k-mer	MAXIMUS	6	5

Tabela 4 - Métricas de avaliação obtidas pelas 17.435 sequências com Rede Neural - DAC.

Algoritmo	Linhagem	Total	Número de Acertos
Rede Neural - DAC	DEL	6	6
Rede Neural - DAC	REINA	6	5
Rede Neural - DAC	ALE	6	6
Rede Neural - DAC	TAR	4	2
Rede Neural - DAC	ANGELA	3	2
Rede Neural - DAC	IVANA	6	6
Rede Neural - DAC	MAXIMUS	6	5

5 Conclusão

Conclui-se que três modelos de algoritmos se destacaram para classificar as linhagens de *Copia* e *Gypsy*: kNN sem PCA DAC, Rede Neural k-mers e Rede Neural DAC. Ao comparar esses três modelos com a predição feita pelo algoritmo Blast, os modelos de aprendizado de máquina se mostraram mais eficientes. Ademais, os modelos de Rede Neural apresentaram indícios de que conseguem classificar eficientemente outros conjuntos de dados de espécies de angiospermas que não foram utilizadas para o treinamento. Os próximos passos desse projeto serão utilizar conjuntos de dados de outras espécies do reino vegetal e até mesmo de outros reinos, a fim de verificar se o modelo adquire boas métricas e consegue fazer uma classificação eficiente.

Referências

ABRUSÁN, G. et al. TEclass - A tool for automated classification of unknown eukaryotic transposable elements. **Bioinformatics**, v. 25, n. 10, p. 1329–1330, 2009.

ALPAYDIN, E. **Introduction to Machine Learning**. [s.l: s.n.]. 2010.

ALTSCHUL, S. F. et al. Basic local alignment search tool**. Journal of Molecular Biology**, v. 215, n. 3, p. 403–410, 1990.

BENNETZEN, J. L.; WANG, H. The contributions of transposable elements to the structure, function, and evolution of plant genomes. **Annual Review of Plant Biology**, v. 65, n. February, p. 505–530, 2014.

CRUZ, E. A. O. et al. Virus-like attachment sites as structural landmarks of plants retrotransposons. **Mobile DNA**, v. 7, n. 1, 2016.

DRECHSLER, F. S. Decision Trees and the Second Law. or, v. 19, n. 4, p. 409, 1968.

FAN, H. et al. Lewin's GENES XI Jones & Bartlett Learning Titles in Biological Science Bioimaging: Current Concepts in Light and Electron Microscopy Biomedical Graduate School: A Planning Guide to the Admissions Process Biomedical Informatics**: A Data Uses Guide Botany.** [s.l: s.n.]. v. 1.

FAWCETT, T. An introduction to ROC analysis. **Pattern Recognition Letters**, v. 27, n. 8, p. 861–874, 2005.

FINNEGAN, D. J. EUKARYOTIC TRANSPOSABLE. **Elsevier Science**. n. 4, 1989.

GEER, John P. Van de. Some Aspects of Minkowsi Distance. [S. l.: s. n.], 1995. 63 p.

URNEY, K. Introduction to neural networks. [s.l: s.n.]. 1997.

IZBICKI, R.; SANTOS T. M. Aprendizado de Máquina - uma abordagem estatística. [s.l: s.n.]. 2020.

KINGMA, D. P.; BA, J. L. Adam: A method for stochastic optimization. 3rd International Conference on Learning Representations, **ICLR 2015 - Conference Track Proceedings**, p. 1–15, 2015.

KLECKNER, N. TRANSPOSABLE ELEMENTS IN PROKARYOTES. **Structure**, p. 341–404, 1981.

LIU, B. et al. Pse-in-One 2.0: An Improved Package of Web Servers for Generating Various Modes of Pseudo Components of DNA, RNA, and Protein Sequences. **Nucleic Acids Research**, v. 43, n. W1, p. W65–W71, 2015.

MARÇAIS, G.; KINGSFORD, C. A fast, lock-free approach for efficient parallel counting of occurrences of k-mers. **Bioinformatics**, v. 27, n. 6, p. 764–770, 2011.

MARTIENSSEN, R. A. Functional genomics: Probing plant gene function and expression with transposons. **Proceedings of the National Academy of Sciences of the United States of America**, v. 95, n. 5, p. 2021–2026, 1998.

MCCARTHY, E. M.; MCDONALD, J. F. LTR STRUC: A novel search and identification program for LTR retrotransposons. **Bioinformatics**, v. 19, n. 3, p. 362–367, 2003.

MCCLINTOCK, B. THE ORIGIN AND BEHAVIOR OF MUTABLE LOCI IN MAIZE. **Proceedings of the National Academy of Sciences**, v. 36, p. 345–355, 1950.

MITA, P.; BOEKE, J. D. How retrotransposons shape genome regulation. **Genetics and Development**, v. 37, p. 90–100, 2016.

MITCHELL, T. M. Machine Learning. [s.l: s.n.]. 1997.

NAKANO, F. K. et al. Top-down strategies for hierarchical classification of transposable elements with neural networks. **International Joint Conference on Neural Networks**, v. 2017- May, p. 2539–2546, 2017.

PARVANDEH, S. et al. Consensus features nested cross-validation. **Bioinformatics**, v. 36, n. 10, p. 3093–3098, 2020.

REZENDE, Solange Oliveira. Sistemas Inteligentes: Fundamentos e Aplicações. 1º. ed. rev. [S. l.]: Editora Manole, 2005. 479 p.

RICHARD, G.-F.; KERREST, A.; DUJON, B. Comparative Genomics and Molecular Dynamics of DNA Repeats in Eukaryotes. **Microbiology and Molecular Biology Reviews**, v. 72, n. 4, p. 686–727, 2008.

SAMUEL, A. L. Some studies in machine learning using the game of checkers. **IBM Journal of Research and Development**, v. 44, n. 1–2, p. 207–219, 1959.

SCHALKOFF, Robert J. Artificial Intelligence: An Engineering Approach. [S. L.]: **McGraw-Hill Education**, 1990. 640 p. 1 v.

THEODORIDIS, S; KONSTANTINOS KOUTROUMBAS**. Pattern Recognition - minder good**. [s.l: s.n.].

SILVA, I. N.; SPATTI, D. H.; FLAUZINO, R. A. Redes Neurais Artificiais Para Engenharia e Ciências Aplicadas. São Paulo: Artliber, 2010.

WICKER, T. et al. A unified classification system for eukaryotic transposable elements. **Nature Reviews Genetics**, v. 8, n. 12, p. 973–982, 2007.