UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
CURSO DE CIÊNCIA DA COMPUTAÇÃO

HENRIQUE VARGAS DAMBROS

Predição de alvos de microRNAs utilizando aprendizado semi-supervisionado e classificação baseada em uma única classe

Monografia apresentada como requisito parcial para a obtenção do grau de Bacharel em Ciência da Computação

Orientador: Profa. Dra. Mariana Recamonde Mendoza

Porto Alegre
2019

AGRADECIMENTOS

Agradeço à Prof.ª Dra. Mariana Recamonde Mendoza, por aceitar ser minha orientadora, função que cumpriu perfeitamente.

Também agradeço à Letícia Dilélio Araujo, ao Fernando Dutra Fagundes Macedo, à minha mãe, Maria Inês Dutra de Vargas, e, novamente, à Prof.ª Dra. Mariana Recamonde Mendoza, por toda a ajuda e o apoio durante esse último e conturbado semestre.

RESUMO

Os microRNAs (miRNAs) são uma classe de RNAs não codificantes, de aproximadamente 22 nucleotídeos, que atuam como silenciadores pós-transcricionais da expressão gênica, inibindo a tradução ou degradando RNA mensageiros (mRNA) alvos. Encontrar pares de miRNAs e mRNAs alvos funcionais é um problema complexo, pois a quantidade de combinações possíveis está na casa dos milhões de pares, dado o elevado número de miRNAs e mRNAs em um organismo, tornando-se inviável avaliar experimentalmente todos os pares existentes. Abordagens utilizando aprendizado de máquina, principalmente no escopo de aprendizado supervisionado, têm sido utilizadas com relativo sucesso na busca por pares de miRNA-mRNA alvo mais prováveis de apresentarem uma interação funcional, possibilitando filtrar os pares a serem testados experimentalmente e auxiliando a descoberta de novos alvos funcionais de miRNAs. No entanto, características dos dados inerentes ao problema, como o grande número de exemplos positivos em contraste aos exemplos negativos, e o amplo volume de interações com fracas evidências experimentais (i.e., com baixa confiança), apresentam desafios ao treinamento de modelos de classificação. A proposta deste trabalho é testar o desempenho de algoritmos de *PU Learning* utilizando pares de miRNA-mRNA com fraca evidência experimental como dados não-rotulados, e de classificação a partir de uma única classe (OCC), especificamente da classe positiva, para o problema de predição de alvos de miRNAs, comparando os resultados com algoritmos supervisionados comumente adotados na literatura relacionada. Os resultados dos experimentos realizados com os dados do Tarbase demonstram que para o *dataset* criado (incluindo nossa definição de dados não-rotulados) e as *features* utilizadas, não há nenhum ganho significativo no desempenho de classificação em se usar métodos de *PU Learning* em relação aos algoritmos supervisionados. No entanto, se não houver dados negativos disponíveis, *PU Learning* se mostra uma opção vantajosa visto que os métodos OCC são muito inferiores em relação aos demais para o problema abordado. Não é possível concluir que os métodos de *PU Learning* não possam ter desempenho melhor que os algoritmos supervisionados na predição de alvos de miRNAs. Diferentes escolhas quanto às *features*, ao conjunto de dados não-rotulados, ou até mesmo do *dataset* utilizado para treinamento dos modelos, podem gerar melhores resultados para estes métodos.

Palavras-chave: Aprendizado de Máquina. PU learning. Classificação Baseada em uma Única Classe. MicroRNA. Predição de Alvos.

MicroRNA Target Prediction based on semi-supervised learning and one-class classification

ABSTRACT

MicrosRNAs (miRNAs) are a family of non-coding RNAs of approximately 22 nucleotides that act as post-transcriptional silencers of gene expression by inhibiting translation or degrading target messenger RNA (mRNA). Finding functional miRNA and target mRNAs pairs is a complex problem, as the number of possible combinations is in the millions of pairs given the high number of miRNAs and mRNAs in an organism, making it impossible to experimentally evaluate all existing pairs. Approaches using machine learning, especially within the scope of supervised learning, have been used with relative success in finding target miRNA-mRNA pairs most likely to have functional interaction, enabling filtering of pairs to be tested experimentally and aiding the discovery of new functional miRNA targets. However, characteristics of the data inherent to the problem, such as the large number of positive as opposed to negative examples, and the large volume of interactions with poor experimental (i.e., low confidence) evidence present challenges to the training of classification models. The purpose of this work is to test the performance of PU Learning algorithms using miRNA-mRNA pairs with poor experimental evidence as unlabelled data, and one-class classification (OCC), specifically of the positive class, for the miRNA target prediction problem, comparing the results with supervised algorithms commonly adopted in the related literature. The results of experiments performed with data from Tarbase demonstrate that for the created dataset (including our definition of unlabelled data) and the features used, there is no significant classification performance gain in using methods of PU Learning in relation to supervised algorithms. However, if no negative data is available, PU Learning proves to be an advantageous option as OCC methods are much worse than others for the problem addressed. It cannot be concluded that PU Learning methods cannot perform better than supervised algorithms in predicting miRNA targets. Different choices of features, unlabeled data, or even datasets used for model training can yield better results for these methods.

Keywords: Machine Learning. PU Learning. One-Class Classification. MicroRNA. Target Prediction.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

DNA	Ácido desoxirribonucleico
miRNA	MicroRNA
mRNA	RNA Mensageiro
PU	*Positive Unlabeled*
OCC	*One-Class Classification*
RNA	Ácido ribonucleico
RNAt	RNA Transportador
SVM	*Support Vector Machine*

SUMÁRIO

1 INTRODUÇÃO

Os microRNAs (miRNAs) são uma classe de RNAs não codificantes, de aproximadamente 22 nucleotídeos, que atuam como silenciadores pós-transcricionais da expressão gênica, inibindo a tradução ou degradando RNA mensageiros (mRNA) alvos. Encontrar pares de miRNAs e mRNAs alvos funcionais é um problema complexo, pois a quantidade de combinações possíveis está na casa dos milhões de pares em razão dos milhares de miRNAs e, especialmente, mRNAs que compõem os organismos (JOHN et al., 2004; BUSHATI; COHEN, 2007; BARTEL, 2009). Portanto, avaliar experimentalmente todos os pares de miRNAs-mRNAs possíveis a fim de definir quais são de fato funcionais (i.e., a ação do miRNA causa a repressão da expressão do mRNA) é infactível devido ao seu alto custo e consumo de tempo (WEN et al., 2018).

O aprendizado de máquina tem se mostrado efetivo em encontrar os pares de miRNAs e mRNAs alvos que apresentam as maiores probabilidades de serem funcionais, isto é, nos quais a ação do miRNA de fato inibe a expressão do mRNA. De posse destas predições, pode-se reduzir o número de interações testadas experimentalmente a fim de buscar alvos de miRNAs funcionais reais sem precisar recorrer a uma escolha aleatória ou busca exaustiva, através das combinações de todos os pares possíveis. Os resultados atuais para predição de alvos de miRNAs através de algoritmos de aprendizado de máquina são bastante promissores, mas existe ainda espaço para melhorias nesta tarefa de classificação.

Para a predição de alvos de miRNAs utilizando aprendizado de máquina, a grande maioria dos trabalhos relacionados utilizam aprendizado supervisionado, como por exemplo *random forests* (MENDOZA et al., 2013), *support vector machine* (SVM) (DING; LI; HU, 2016) e, mais recentemente, *deep learning* (WEN et al., 2018). No entanto, um desafio enfrentado pelos trabalhos relacionados é a pouca quantidade de exemplos de pares miRNAs-mRNAs não-funcionais presentes em bases de dados de alvos de miRNAs validados experimentalmente, como o DIANA-Tarbase V7.0 (VLACHOS et al., 2014). Como consequência, existe um significativo desbalanceamento de exemplos positivos (funcionais) e negativos (não-funcionais) para treinamento dos métodos de aprendizado. Este desbalanceamento de dados apresenta-se como um problema para a grande maioria dos algoritmos de aprendizado supervisionado, pois degrada o desempenho dos modelos ao introduzir um viés de aprendizado para a classe majoritária, ignorando a classe minoritária (GUO et al., 2008). Adicionalmente, existe um grande número de exemplos positivos de

miRNA-mRNAs alvos suportado por evidência experimental fraca, e.g., técnicas que não permitem afirmar a existência de uma interação direta entre as sequências, podendo este efeito ser derivado de uma interação indireta) (VLACHOS et al., 2014). Estes exemplos apresentam baixa confiança no papel regulatório do miRNA sobre um alvo específico. Assim, não é claro se estes pares são de fato funcionais e se podem ser considerados como exemplos positivos na tarefa de aprendizado.

A proposta deste trabalho é abordar o problema de predição de alvos de miRNAs através da utilização de métodos de *PU Learning* e de classificação a partir de uma classe (OCC). Estes métodos possuem a vantagem de aprenderem a partir de dados rotulados positivos (em complemento a dados não rotulados, no caso de *PU Learning*), sem precisar de dados negativos rotulados. O objetivo é investigar se estes métodos terão um desempenho de classificação melhor ou comparável aos algoritmos de aprendizado supervisionado, por não utilizarem dados rotulados negativos e não serem afetados pelo desbalanceamento das classes positiva e negativa, presente nas bases de dados de pares de miRNAs e mRNAs alvos experimentalmente validados. Nossos resultados demonstram que para o *dataset* criado e as *features* utilizadas, não há nenhum ganho de desempenho significativo em se usar métodos de *PU Learning* em relação aos algoritmos supervisionados, e que os métodos OCC são muito inferiores em relação aos demais (i.e., algoritmos supervisionados e de *PU Learning*) para o problema abordado. No entanto, não é possível concluir que os métodos de *PU Learning* não possam ter desempenho melhor que os algoritmos supervisionados na predição de alvos de miRNAs. É possível que uma escolha melhor das *features*, do conjunto de dados não-rotulados, ou até mesmo do *dataset* utilizado para treinamento dos modelos, possam gerar melhores resultados para estes métodos.

O trabalho está organizado como segue: O Capítulo 2 apresenta o conhecimento teórico necessário para entender os Capítulos seguintes; o Capítulo 3 apresenta os trabalhos relacionados da literatura; o Capítulo 4 apresenta a metodologia do trabalho; o Capítulo 5 apresenta os experimentos e resultados; e o Capítulo 6 apresenta as conclusões e propostas para trabalhos futuros.

2 REFERENCIAL TEÓRICO

Neste Capítulo são apresentados os principais conceitos relacionados ao trabalho, formando o embasamento teórico necessário para compreensão dos Capítulos seguintes.

2.1 Dogma Central da Biologia Molecular

Conforme Voet and Voet (2013), a determinação da estrutura do DNA por Watson e Crick em 1953, é frequentemente referida como marco do nascimento da biologia molecular. A estrutura de Watson – Crick do DNA é importante, pois além de fornecer a estrutura que pode ser considerada como o centro da vida, sugeriu o mecanismo da hereditariedade. Em 1958, Crick resumiu as linhas gerais do processo da expressão gênica e replicação, onde o DNA dirige a sua própria replicação e sua transcrição para produzir o RNA, o qual por sua vez, dirige a sua tradução para a formação de proteínas. Uma fita de DNA dirige a síntese de RNA, através do processo denominado de transcrição. Os RNA mensageiros (mRNA) orientam essa síntese proteica, estrutura cuja função principal é a manifestação das características hereditárias, contidas no DNA (ZAHA; FERREIRA; PASSAGLIA, 2014).

De acordo com CAMPBELL and FARREL (2001), na síntese de proteínas, processo denominado de tradução gênica, além do mRNA, participam ainda, vários RNAt (RNA transportador), um ribossomo e aminoácidos. Uma série de segmentos consecutivos de três nucleotídeos, são conhecidos como códons, e cada códon determina um aminoácido específico. O processo de transcrição é mais complexo em eucariotos do que em procariotos. Existem três RNA polimerases com diferentes atividades, são elas:

1. RNA – polimerase I, que sintetiza a maioria dos precursores de RNA's ribossômicos;
2. RNA – polimerase II sintetiza os precursores de mRNA;
3. RNA – polimerase III sintetiza os RNAts).

A correspondência entre a sequência de bases em um códon e o resíduo de aminoácido que ele específica é conhecida como código genético (CAMPBELL; FARREL, 2001).

2.2 MicroRNAs

A partir do ano 2000, o interesse científico em relação aos microRNAs (miRNAs) vem aumentando gradativamente, em razão da crescente evidência do envolvimento destas moléculas no funcionamento dos organismos e no desenvolvimento e progressão de doenças. MicroRNAs são pequenos RNAs (aproximadamente 22 nucleotídeos) não codificantes, capazes de modular a expressão gênica em nível pós-transcricional, inibindo a expressão do gene através da degradação do transcrito ou do bloqueio da sua tradução para uma proteína (WANG et al., 2010).

Um dos critérios para que ocorra a interação entre o miRNA e mRNA, é a complementaridade entre os nucleotídeos de ambas as sequências. Cada miRNA liga-se a muitos RNA mensageiros, e estes podem ter sua estabilidade ou tradução definida por mais de um microRNA, cuja característica os tornam um mecanismo eficiente para associação a doenças e processos biológicos (ALVES, 2016).

2.3 Aprendizado de Máquina

O aprendizado de máquina é uma das subáreas da inteligência artificial, com foco em desenvolver métodos, técnicas e ferramentas sobre o aprendizado, capaz de aprender e predizer sobre um grande volume de dados informados e adquirir conhecimento de forma automática. Esses programas são capazes de tomar decisões, a partir de experiências acumuladas de soluções que obtiveram sucesso em testes anteriores (MONARD; BARANAUSKAS, 2003). Para Mitchell et al. (1997), "um programa aprende a partir da experiência E, em relação a uma classe de tarefas T, com uma medida de desempenho P, se seu desempenho em T, medido por P, melhora com E".

As técnicas de aprendizado de máquina empregam um meio de inferência denominado indução, no qual obtém-se conclusões genéricas a partir de um conjunto de exemplos. O aprendizado indutivo pode ser de dois tipos principais: supervisionados e não-supervisionados. Enquanto o aprendizado supervisionado necessita de dados rotulados, o não-supervisionado analisa somente as *features* para detectar os padrões implícitos aos dados. Há ainda, um terceiro tipo de aprendizado de máquina utilizado nesse trabalho, o aprendizado semi-supervisionado, que fica entre os aprendizados supervisionado e não-supervisionado. Esta técnica utiliza tanto dados rotulados (supervisionado) quanto os dados não-rotulados (não-supervisionado). A seguir serão discutidos os métodos super-

visionados, não-supervisionados e semi-supervisionados utilizados nesse trabalho.

2.3.1 Aprendizado Supervisionado

No aprendizado supervisionado, o *dataset* é composto por entradas e saídas. Cada entrada é um dado de exemplo e possui valores para uma ou mais *features*, e a saída é a resposta esperada do algoritmo para a respectiva entrada. O objetivo do algoritmo é aprender uma generalização dos dados e produzir saídas razoáveis, para entradas não presentes nos dados utilizados para o treinamento do algoritmo, denominadas dados de teste (MARSLAND, 2014).

Os métodos de aprendizado supervisionado podem ser divididos em dois tipos: regressores e classificadores. Os regressores geram valores contínuos como saída, e, os classificadores geram rótulos de classe como saída. Nesse trabalho serão utilizados classificadores.

Os classificadores também podem ser subdivididos em dois tipos: classificação binária e multi-classe. Na classificação binária existem somente dois rótulos de classe possíveis. Já na classificação multi-classe deve existir mais do que dois rótulos. Nesse trabalho, a classificação binária é a utilizada.

Abaixo são listados os algoritmos supervisionados utilizados nesse trabalho, com uma breve descrição do seu funcionamento.

- *Árvore de Decisão*: A árvore de decisão é um algoritmo de classificação que cria uma árvore binária, em que cada nodo interno é uma pergunta do tipo "*if-then-else*", baseado no valor de uma *feature*, e os nodos folhas são associados às classes. A escolha da *feature* a ser usada e a escolha do valor de separação do nodo são baseadas em um processo de medida de entropia de informação. O objetivo, nesse caso, é recursivamente dividir os dados de entrada, reduzindo ao máximo a medida de impureza adotada, através da escolha de uma *feature* e um valor que separem os dados (MARSLAND, 2014). Nesse trabalho, a medida de impureza *Gini Index* foi usada.
- *Random Forest*: O método *Random Forest* (RF), proposto por Breiman (2001), é um algoritmo de classificação que utiliza um comitê (*ensemble*) de árvores de decisão. Cada árvore de decisão recebe uma amostra, com reposição, dos dados de entrada. Além disso, em cada nodo as árvores tem acesso somente a um subcon-

junto das *features* para serem utilizadas para a divisão dos dados, o qual é escolhido aleatoriamente a fim de aumentar a diversidade entre as árvores. A classificação final do RF é uma média da classificação das árvores utilizadas. O voto majoritário é utilizado no caso de predições em forma de rótulo e, a média aritmética é utilizada no caso de predições em forma de valores contínuos.

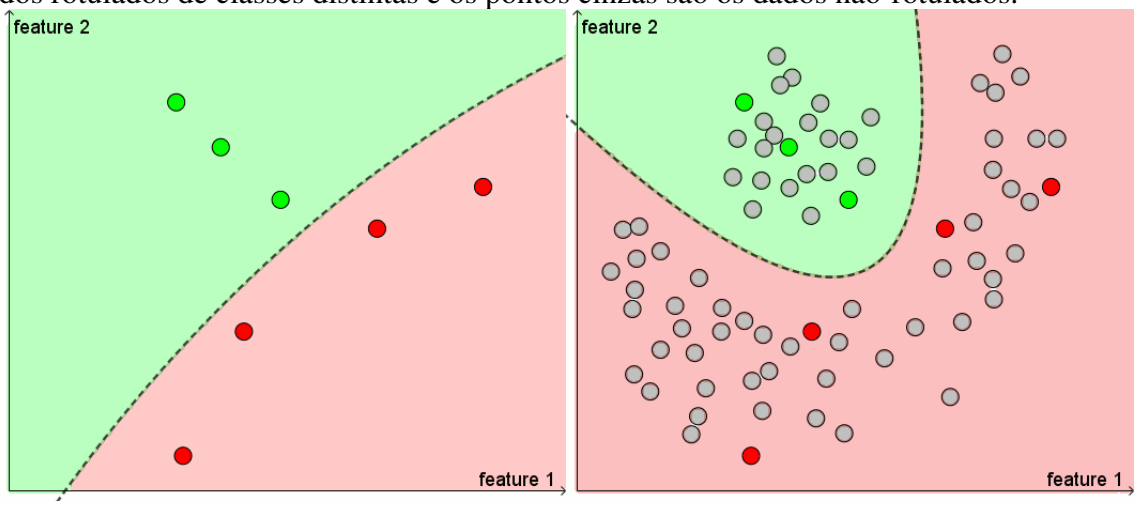
- *Support Vector Machine*: Segundo Marsland (2014), as máquinas de vetores de suporte (SVM, de *support vector machine*) compõem um conjunto de métodos do aprendizado supervisionado que utilizam o método Kernel, proposto por Vapnik (1995), para fazer uma transformação não-linear dos dados, criando uma quantidade maior de dimensões, visando possibilitar uma separação linear dos dados a partir de hiperplanos. O objetivo final é encontrar os hiperplanos que criem uma separação ótima das classes. Sendo a separação ótima aquela na qual o hiperplano tem a maior distância possível para as instâncias mais próximas de cada classe.
- *Regressão Logística*: Regressão Logística é um modelo linear de classificação. Esse método utiliza uma função sigmoide logística sobre uma função linear, para gerar valores de probabilidade para as classes (BISHOP, 2006).

2.3.2 Aprendizado Semi-Supervisionado e *PU learning*

No aprendizado semi-supervisionado, os dados não-rotulados são usados no treinamento junto aos dados rotulados usuais do aprendizado supervisionado. Esse tipo de aprendizado é especialmente útil quando o custo de obtenção de novos dados é baixo, porém o custo de rotular esses dados é alto, gerando, portanto, uma grande disponibilidade de dados com apenas uma pequena porcentagem desses dados possuindo rótulos. O objetivo do aprendizado semi-supervisionado é gerar uma classificação mais precisa quando comparada a um algoritmo supervisionado, ao extrair informações extras dos dados não-rotulados (CHAPELLE; SCHLKOPF; ZIEN, 2010). A Figura 2.1 mostra uma representação do possível ganho na classificação ao utilizar os dados não-rotulados.

Positive-unlabeled learning, ou *PU learning*, é um tipo de aprendizado semi-supervisionado de classificação binária em que os dados rotulados são constituídos somente pela classe positiva (**P**) e os dados não-rotulados (**U**) possuem dados que podem ser tanto da classe positiva quanto da classe negativa (ELKAN; NOTO, 2008). Neste trabalho foram utilizados dois métodos de *PU learning*: *Two-Step* e *PU bagging*, ambos

Figura 2.1: A esquerda um exemplo de classificação supervisionada e a direita um exemplo de classificação semi-supervisionada. Os pontos verdes e vermelhos representam dados rotulados de classes distintas e os pontos cinzas são os dados não-rotulados.



Fonte:(WRIGHT, 2017)

descritos abaixo.

2.3.2.1 Two-Step

Esse método é dividido em dois passos:

- No primeiro passo, uma técnica é utilizada para extrair o conjunto **RN** (*"reliable negatives"*), o qual é um subconjunto de **U** e possui exemplos considerados como negativos com um alto grau de confiança.
- No segundo passo, um algoritmo de classificação binária é treinado sobre os dados positivos e o conjunto **RN** para gerar a classificação final dos dados. Este segundo passo também pode ser executado iterativamente, adicionando os dados do conjunto **U** classificados como negativos ao conjunto **RN** e repetindo a execução desse passo.

Em Kaboutari, Bagherzadeh and Kheradmand (2014), várias técnicas de extração do conjunto **RN** são descritas como *Spy*, *Rocchio* e *1DNF*. No entanto, qualquer técnica que gere um conjunto de negativos a partir do conjunto **U** pode ser utilizada. A técnica utilizada neste trabalho, primeiramente, treina um algoritmo supervisionado sobre os conjuntos **P** e **U**, considerando **U** como a classe negativa. Após o treinamento, a classificação é executada sobre todos os dados atribuindo valores em forma de probabilidade de cada exemplo ser da classe positiva e, a partir desta classificação, são considerados *reliable negatives* os dados do conjunto **U** que forem atribuídos valores de probabilidade inferior ao exemplo positivo que recebeu a menor probabilidade, em relação a todos os outros positivos.

2.3.2.2 PU Bagging

Este método é uma adaptação do algoritmo *bagging* de classificação supervisionada ao problema de *PU learning*. Mordelet and Vert (2014) define o método como:

- Treine vários classificadores binários utilizando o conjunto **P** como positivo e uma amostra (diferente para cada classificador) do conjunto **U** com reposição como negativo.
- A função de avaliação final será a média das funções de avaliação de cada um dos classificadores.

O algoritmo *PU bagging* também resolve o problema comum de desbalanceamento dos dados dos conjuntos **P** e **U**, em que o conjunto **P** é muito menor do que **U**, ao criar amostras do conjunto **U** para o treinamento dos classificadores. O resultado é que cada classificador é treinado utilizando um conjunto de treino com quantidades iguais de **P** e **U**.

2.3.3 *One-Class Classification*

O objetivo da classificação de classe única (OCC, de *One-Class Classification*) é reconhecer dados de uma determinada classe (normais) e rejeitar dados de todas as outras classes (anormais ou *outliers*). Nesse tipo de classificação não é necessário ter exemplos de dados que não são da classe alvo (KHAN; MADDEN, 2009).

O OCC, também chamado de *anomaly detection*, pode ser dividido em dois métodos: *outlier detection* e *novelty detection*.

- *Outlier detection* é um método não-supervisionado, em que os dados de treinamento podem conter *outliers*.
- *Novelty detection* é um método semi-supervisionado, em que os dados de treinamento devem conter somente os dados normais. Além disso, os *outliers*, nesse caso chamados de *novelties*, podem formar *clusters*, desde que estejam em uma região de baixa densidade dos dados de treinamento (normais).

Neste trabalho dois métodos OCC foram utilizados: *One-Class* SVM e *Isolation Forest*, ambos descritos abaixo.

2.3.3.1 One-class SVM

O *one-class* SVM tenta separar a região que contém dados, da região que não contém dados. Isso é realizado construindo um hiperplano que possui uma distância máxima da origem, mas mantendo todos os dados do lado oposto a origem, na separação do hiperplano (KHAN; MADDEN, 2009).

Esse método pode ser treinado em modo *outlier detection* ou *novelty detection*.

2.3.3.2 Isolation Forest

Segundo Liu, Ting and Zhou (2008), *Isolation Forest* é um método *ensemble* de classificadores do tipo árvore. Cada árvore é construída escolhendo uma *feature* e um valor aleatório, entre o mínimo e máximo desta *feature*, e separando os dados baseado nesse valor. Esse passo é repetido até que todos os dados estejam isolados em um nó folha.

Essa separação aleatória produz profundidades visivelmente menores para dados anômalos. Isso acontece porque (a) instâncias com valores de *features* distinguíveis têm uma chance maior de serem separadas no início da construção da árvore, e (b) as menores quantidades de anomalias geram um menor número de particionamentos e, portanto, menor profundidade.

A função de decisão utilizada para classificar as instâncias é a média das profundidades geradas por todas as árvores. Quando uma floresta gera, na média, uma profundidade pequena para instâncias específicas, há uma grande chance destas serem anômalas. O método *isolation forest*, assim como o *one-class SVM*, também pode ser treinado em modo *outlier detection* ou *novelty detection*.

2.3.4 Normalização das entradas por padronização

Na normalização, os intervalos originais dos valores das *features* são transformados para outro intervalo específico, usualmente igualando o intervalo para as diferentes *features*. A normalização deve ser aplicada a cada *feature*, individualmente, e pode ser por duas formas: amplitude (por reescala ou por padronização) ou por distribuição. Neste trabalho a normalização utilizada é a de padronização.

Na padronização é definido um valor central e de espelhamento comuns para todos as *features*, e consiste em subtrair a média dos valores de uma *feature*, dividindo o

resultado pelo desvio padrão. Após feita esta normalização, tem-se que a média dos novos dados, para cada *feature*, ficará centrada em 0 e a variância será 1 (PEDREGOSA et al., 2019b).

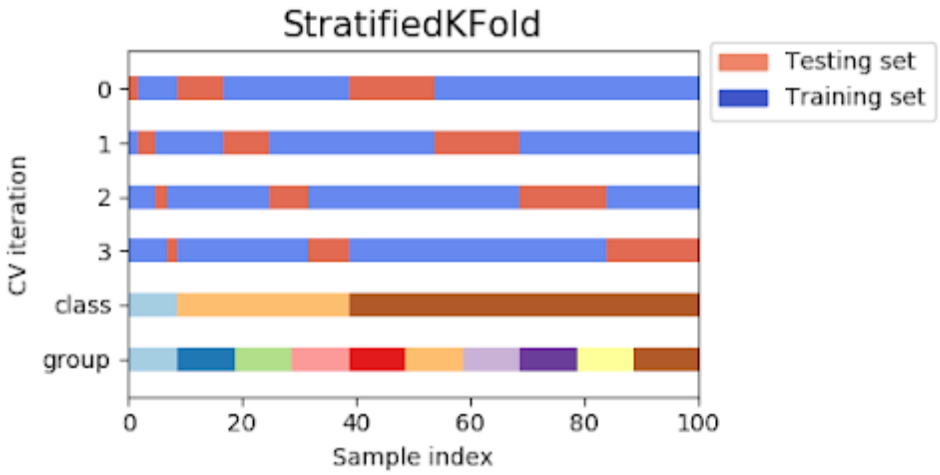
2.3.5 Validação Cruzada

A validação cruzada *k-fold*, ou *k-fold cross-validation*, é uma técnica de validação de modelos, para verificar se o algoritmo de aprendizado de máquina não possui *overfitting*, bem como avaliar o desempenho do algoritmo para dados fora do conjunto de dados de treinamento.

O método de validação cruzada *k-fold*, consiste em dividir o conjunto total de dados em k subconjuntos disjuntos de mesmo tamanho. Um desses subconjuntos é utilizado para teste e os k-1 restantes são utilizados para treino, repetindo k vezes, de forma a alternar o subconjunto de teste, a final, apura-se a média das performances de validação dos k experimentos, obtendo assim, uma estimação de erro de generalização. A variância da estimativa do erro de validação diminui conforme k aumenta.

O método utilizado nesse trabalho é o *k-fold* estratificado. Este método tem, como única diferença, em relação ao método *k-fold*, levar em conta as proporções de cada classe e mantê-las nos k-subconjuntos gerados. A Figura 2.2 apresenta um exemplo desse tipo de validação cruzada.

Figura 2.2: Visualização do método de validação cruzada *k-fold* estratificado, para k=4.



Fonte: (PEDREGOSA et al., 2019a)

2.3.6 Métricas de avaliação de desempenho

No presente trabalho, a avaliação de desempenho foi realizada a partir da análise da matriz de confusão, também conhecida como tabela de confusão, que consiste em uma tabela ou uma matriz de dimensão 2X2 comparando os valores reais e preditos para cada exemplo de teste. É uma métrica voltada para modelos de classificação e, seu objetivo é calcular a quantidade de **falsos positivos** e **falsos negativos**; bem como **verdadeiros positivos** e **verdadeiros negativos**.

- Verdadeiro positivo (**TP**): previsão positiva correta
- Falso positivo (**FP**): previsão positiva incorreta para um exemplo da classe negativa
- Verdadeiro negativo (**TN**): previsão negativa correta
- Falso negativo (**FN**): previsão negativa incorreta para um exemplo da classe positiva

Figura 2.3: Matriz de confusão.

		valor predito	
		positivo	negativo
valor verdadeiro	positivo	TP	FN
	negativo	FP	TN

Fonte: Autor

Reunindo as 4 métricas acima, damos origem à matriz de confusão, ilustrada na Figura 2.3. A partir desta matriz, podem ser extraídas inúmeras métricas de desempenho, algumas das quais serão definidas a seguir.

2.3.6.1 Sensibilidade e Especificidade

Sensibilidade ou *recall*: é a capacidade do modelo em predizer corretamente a classe de instâncias positivas, refletindo o seu potencial em identificar a proporção de verdadeiros positivos (predições positivas corretas), em relação ao total de positivos da amostra. Seu valor varia entre 0 e 1, e seu cálculo é definido por:

$$sensibilidade = \frac{TP}{TP + FN} \tag{2.1}$$

De acordo com a fórmula acima (2.1), a sensibilidade é a razão da quantidade de amostras classificadas corretamente para a classe positiva (TP), dividido pelo total de amostras que pertencem a esta classe (TP + FN).

Especificidade (*specificity*): é a proporção de casos negativos que foram identificados corretamente na amostra, ou seja, é a taxa de verdadeiros negativos. Portanto, a especificidade pode ser definida como a capacidade do modelo em predizer corretamente, que determinada instância pertence a classe negativa. A fórmula da especificidade está descrita abaixo (2.2).

$$especificidade = \frac{TN}{TN + FP} \tag{2.2}$$

2.3.6.2 Curvas ROC e a Área sob a curva (AUC)

A curva ROC (*Receiver Operating Characteristic*) é uma técnica gráfica utilizada para avaliar a capacidade de um método de aprendizado. Os classificadores utilizados nesse trabalho geram classificações binárias, em forma de probabilidade preditas para a classe positiva. A curva ROC permite fazer análises visuais entre sensibilidade e especificidade, relativamente a diversos pontos de corte (em inglês *tresholds*) das probabilidades geradas. O ponto de corte é definido como um valor entre 0 e 1, acima desse valor o exemplo é classificado como positivo e abaixo desse valor é classificado como negativo.

No gráfico da curva ROC, os eixos são definidos por convenção da seguinte forma: 1-especificidade representada no eixo X e a sensibilidade é representada no eixo Y, ambas variando de 0 a 1 (0-100%). Quanto mais próxima a curva estiver do canto superior esquerdo do espaço destinado a representação gráfica, melhor será o desempenho do método avaliado.

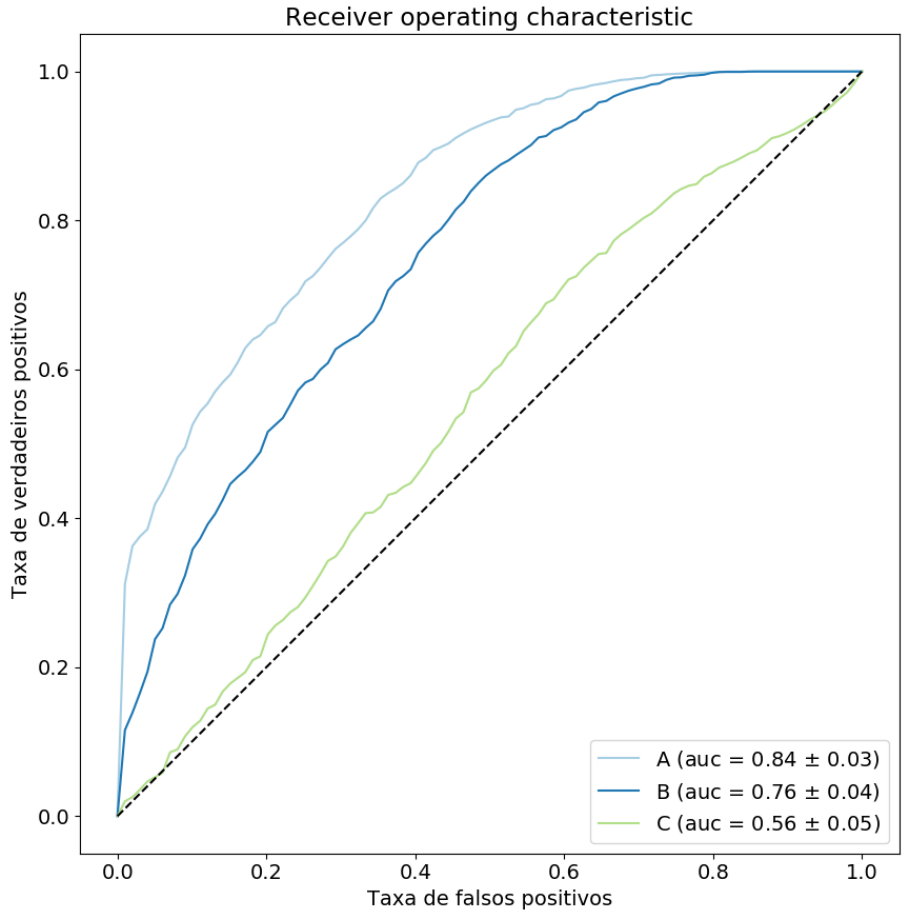
Outra nomenclatura para os parâmetros da curva ROC é a taxa de verdadeiros positivos no eixo X e a taxa de falsos positivos no eixo Y. As definições dessas métricas são:

- Taxa de verdadeiros positivos (True Positive Rate): TP / (TP + FN)
- Taxa de falsos positivos (False Positive Rate): FP / (FP + TN)

Uma métrica derivada da curva ROC utilizada neste trabalho é a área sob a curva (AUC). Sua representação se faz de forma numérica, no intervalo de 0 a 1, e representa o valor da área abaixo da curva ROC. Quando o valor se aproxima de 1, nesse caso, melhor será o desempenho do classificador e, quando ocorrer a proximidade da diagonal, está

implicará em uma área em torno de 0,5, o que demonstra uma predição aleatória. Um exemplo do gráfico da curva ROC com valores de AUC na legenda é demonstrado na Figura 2.4.

Figura 2.4: Exemplo de gráfico da curva ROC.



Fonte: Autor

2.3.7 Redução de Dimensionalidade

Segundo Marsland (2014), algoritmos de redução de dimensionalidade transformam os dados para um espaço de dimensionalidade menor, ou seja, reduzem a quantidade de *features* do *dataset*. Existem dois benefícios principais da redução de dimensionalidade:

- Possibilitar a visualização de dados de alta dimensionalidade ao reduzir o espaço para duas ou três dimensões, facilitando assim a interpretação dos dados e dos resultados gerados pelos algoritmos de aprendizado de máquina.
- Impactar positivamente o desempenho de algoritmos de aprendizado de máquina,

por causa da chamada maldição de dimensionalidade (*curse of dimensionality*), que refere-se a todos os problemas que aparecem quando utiliza-se dados de alta-dimensionalidade.

Neste trabalho a redução de dimensionalidade foi usada para a visualização dos dados. Os algoritmos de redução de dimensionalidade utilizados foram:

- *Principal Component Analysis (PCA)*: Algoritmo linear e não-supervisionado de redução de dimensionalidade. Este algoritmo gera uma transformação das *features* originais para um novo conjunto de *features* que retém a maior variância nos dados (JOLLIFFE, 2011).
- *Isomap*: Algoritmo não-linear e não-supervisionado de redução de dimensionalidade. È uma variação do algoritmo *Multi-Dimensional Scaling* (MDS). Isomap projeta os dados para um espaço de dimensionalidade menor tentando manter as mesmas distâncias entre os pontos, essa distância entre os pontos é calculada a partir do menor caminho em um grafo de vizinhança (MARSLAND, 2014; SILVA; TENENBAUM, 2003).
- *Neighbourhood Components Analysis (NCA)*: Algoritmo linear e supervisionado de redução de dimensionalidade. Aprende um transformação linear dos dados que maximiza o desempenho médio de um variante estocástico do algoritmo de classificação *leave-one-out K-Nearest Neighbors* (GOLDBERGER et al., 2005).
- *T-distributed Stochastic Neighbor Embedding (T-SNE)*: Algoritmo não-linear e não-supervisionado de redução de dimensionalidade. Modela as distâncias entre pontos no espaço de dimensionalidade menor a valores de similaridade calculados a partir de uma distribuição *student t-test*. Contém parâmetros que que podem ser modificados para gerar diferentes visualizações. Estes são: número de iterações, taxa de aprendizado e perplexidade (*perplexity*) (MAATEN; HINTON, 2008).

3 TRABALHOS RELACIONADOS

A predição de alvos de miRNAs através de algoritmos de aprendizado de máquina já é bastante estudada na literatura. No entanto, a maioria dos trabalhos relacionados utilizam métodos supervisionados. Nas seções abaixo, são descritos os trabalhos da área utilizando algoritmos OCC e *PU learning*, assim como alguns trabalhos mais novos da área de predição de alvos de miRNA que utilizam aprendizado supervisionado. Ressalta-se que os esforços baseados em métodos de aprendizado semi-supervisionado na tarefa de predição de alvos de miRNAs ainda são bastante escassos, justificando o pequeno número de artigos relacionados revisados pelo presente trabalho.

3.1 *One-Class Classification*

A utilização de algoritmos OCC para problemas da área de miRNAs é estudada em Yousef et al. (2008); Yousef, Najami and Khalifav (2010) e Yousef, Allmer and Khalifa (2016). Em Yousef, Najami and Khalifav (2010) e Yousef, Allmer and Khalifa (2016), métodos de OCC são utilizados para a predição de alvos de miRNAs e sua performance é comparada aos resultados de algoritmos supervisionados. Yousef, Allmer and Khalifa (2016) estuda especificamente a seleção de *features*, para previsão de alvos de miRNAs utilizando algoritmos OCC. Em Yousef et al. (2008), algoritmos OCC são utilizados, para a identificação de genes de miRNAs que é um outro problema da área de miRNAs. Nos três trabalhos, um dos algoritmos utilizados é o *One-Class* SVM, também utilizado neste trabalho. Os resultados dos trabalhos mencionados acima apontam valores de acurácia, sensibilidade e especificidade de 99% para alguns algoritmos supervisionados, o que indicaria uma classificação praticamente perfeita. Estes valores, como comentado também na Seção 5.1, provavelmente não são indicativos de um desempenho real desses algoritmos, visto que diante da alta complexidade do problema de predição torna-se improvável uma classificação perfeita. Portanto, torna-se difícil tirar conclusões a partir destes resultados, tendo em vista a alta chance de viés no treinamento dos modelos reportados pelos autores.

3.2 *PU learning*

Em Pio et al. (2014), é proposto um método de *PU learning* de predição de alvos de miRNAs para a descoberta de redes de regulação genética guiadas por miRNAs. O método proposto utiliza os *scores* gerados por outros classificadores de predição de alvos de miRNAs disponíveis na literatura como *features* do seu *dataset*. A partir dos dados positivos e não-rotulados, é treinado um comitê de classificadores SVM para gerar probabilidades de os dados serem rotulados (positivos). Com base nestes valores de probabilidade, é treinado outro comitê de classificadores SVM, cujas probabilidades preditas são utilizadas para gerar rótulos e pesos para os dados não-rotulados. Os rótulos são designados como positivos se a probabilidade é maior que 0,5 e negativos se menor que 0,5. Os pesos são maiores e influenciam mais a classificação quando a probabilidade gerada é mais próxima de 0 ou 1, e menores quando estas probabilidades se aproximam de 0,5. O método descrito acima é um método de *PU learning* baseado em pesos e é diferente dos dois métodos de *PU learning* utilizados no presente trabalho; *two-step* e *PUBagging*.

3.3 *RLSMDA*

Chen and Yan (2014) desenvolveram um método que chamaram *Regularized Least Squares for MiRNA-Disease Association* (RLSMDA), que é um método semi-supervisionado que utiliza somente os dados positivos como rotulados. O artigo não entra em detalhes na implementação, então, não fica claro se ele é um algoritmo OCC ou *PU learning*. O método é utilizado para encontrar associações de miRNAs a doenças, se diferenciando portanto do objetivo do presente estudo.

3.4 Métodos supervisionados

Métodos de aprendizado utilizando redes neurais profundas estão em alta, em diversas áreas que utilizam aprendizado de máquina e, isso não é diferente na predição de alvos de miRNAs. Três trabalhos recentes utilizam aprendizado profundo. Cheng et al. (2015) propõem um classificador utilizando uma rede neural convolucional, Lee et al. (2016) propõem um classificador utilizando uma rede neural recorrente e, Wen et al. (2018) propõem um classificador utilizando um *autoencoder*, que é baseado em uma

rede neural recorrente. Enfatizamos, no entanto, que todos estes trabalhos dependem da existência de dados rotulados para ambas as classes, positivas (pares miRNA-mRNAs funcionais) e negativas (pares miRNA-mRNAs não funcionais).

3.5 Comparação de Resultados

Cada trabalho de predição de alvos de miRNAs utiliza um dataset próprio, gerando valores de desempenho de classificação específicos para o dataset utilizado. Em razão disso, não é possível comparar diretamente os resultados dos diferentes trabalhos relacionados entre si, assim como não é possível compará-los diretamente aos resultados do presente trabalho. Para poder comparar diretamente o desempenho de classificação dos métodos de PU Learning e OCC, utilizados neste trabalho, com os algoritmos supervisionados, utilizados nos trabalhos relacionados, foi necessário implementar os algoritmos supervisionados e avaliar seu desempenho para o dataset utilizado, juntamente com a avaliação dos métodos de PU Learning e OCC.

4 ABORDAGEM PROPOSTA

A proposta deste trabalho é avaliar o desempenho de algoritmos OCC e *PU learning*, quando comparados ao desempenho de algoritmos supervisionados para a predição de alvos de miRNAs. Nas próximas seções será explicado em detalhes cada componente da abordagem proposta.

4.1 Fontes de Dados

4.1.1 DIANA-TarBase

A base de dados DIANA-Tarbase V7.0 (VLACHOS et al., 2014), ou simplesmente Tarbase, é uma base de dados que contém dados de interação de pares de miRNAs e mRNAs. Esses dados são curados manualmente e proveem da análise de centenas de milhares de publicações da literatura que contém dados experimentalmente validados.

Os dados podem ser divididos entre positivos, ou seja, em que o miRNA interage com o mRNA, e negativos, em que o miRNA não interage com o mRNA. Um problema que se tem com relação a esses dados é que a quantidade de exemplos negativos é muito menor que a quantidade de exemplos positivos. Em parte, este problema deriva da dificuldade de se publicar estudos negativos, isto é, estudos que não validam a hipótese de que um mRNA é regulado por um dado miRNA. Esse desbalanceamento dos dados foi a motivação para o uso de algoritmos de *PU learning* e OCC, que não usam dados rotulados da classe negativa e, portanto, em teoria não são afetados pelo desbalanceamento das classes.

Além da divisão de dados em positivos e negativos, mais duas divisões foram utilizadas de acordo com a natureza da regulação e o tipo de validação.

- A regulação separa os dados entre UP e DOWN. UP representa as interações em que o miRNA aumenta a síntese de proteína do mRNA e DOWN representa interações em que a síntese diminui. As interações DOWN são as que interessam a esse trabalho, portanto, os dados que possuíam interação classificada como UP foram descartados.
- O tipo de validação separa os dados entre direta e indireta. Os dados com validação direta possuem uma evidência mais forte de interação física entre os pares miRNA

e mRNA, portanto, foram utilizados para gerar os dados rotulados. Os dados com validação indireta, que significam uma evidência fraca, foram utilizados para a geração de dados não-rotulados.

Os dados provenientes dos métodos de validação CLIP-Seq (i.e., *HITS-CLIP* e *PAR-CLIP*), foram considerados nesse trabalho como evidência fraca, visto que apresentam confiança limitada acerca da efetividade da ação do miRNA sobre o seu alvo (VLACHOS et al., 2014). Portanto, foram utilizados para a geração dos dados não-rotulados, juntamente com os dados de validação indireta. Além destes dados, os exemplos de pares miRNAs-mRNAs que apareciam duplicados nos dados positivos e negativos também foram incluídos na geração de dados não-rotulados, tendo em vista que as evidências são inconclusivas.

Na Tabela 4.1, pode-se ver a divisão dos dados pelos seus diferentes tipos e as quantidades de exemplos de pares para cada tipo.

Tabela 4.1: Dados utilizados da base Tarbase, divididos por tipo.

	Positivos Diretos	Negativos Diretos	Indiretos	HITS_CLIP_PAR_CLIP	Duplicados	Total
Quantidade	5.619	1.944	23.335	278.541	17	309.456

Fonte: Autor

4.1.2 Sequências de microRNA e mRNA

A base de dados Tarbase prove uma lista de pares de miRNAs e mRNAs representados por seus identificadores. No entanto, para extrair as *features* para a predição de alvos são necessárias as sequências de nucleotídeos dos miRNAs e mRNAs, as quais não estão contempladas no Tarbase. As sequências de miRNAs foram coletadas a partir da base de dados MirBase (GRIFFITHS-JONES et al., 2006; KOZOMARA; GRIFFITHS-JONES, 2013), enquanto que as sequências de mRNAs foram coletadas do BioMart portal (SMEDLEY et al., 2015), uma interface para acessar mais de 800 bases de dados diferentes.

Nos dados de sequências de mRNAs, um mesmo mRNA, representado pelo seu identificador, pode ter várias sequências correspondentes. Isso decorre de aspectos da transcrição do DNA, sendo que cada uma dessas sequências diferentes é chamada de transcrito. Nesse trabalho, todos os transcritos associados a cada mRNA foram utilizados.

4.2 Geração do *Dataset*

A geração do *dataset* foi baseada na metodologia utilizada no trabalho de Moita (2018). No entanto, diferentemente do trabalho anterior, o presente estudo prevê a criação de um conjunto de dados não-rotulados. As etapas na construção do *dataset* são descritas a seguir.

Primeiramente, os pares de miRNA e mRNA provenientes do Tarbase foram separados entre positivos, negativos e não-rotulados. Os positivos compreendem os dados positivos de validação direta e os negativos compreendem os dados negativos de validação direta. Os não-rotulados compreendem os dados de validação indireta, os dados do método de validação *HITS-CLIP* e *PAR-CLIP*, e os exemplos de pares que apareceram duplicados nos dados positivos e negativos. O total de exemplos gerados para cada conjunto foi: **5.619** exemplos positivos, **1.944** exemplos negativos e **301.893** exemplos não-rotulados. Ressaltamos que no trabalho anterior, não foram utilizados os dados não-rotulados.

Na passo seguinte, esses pares de miRNA e mRNA foram transformados em pares de sequências de nucleotídeos. Devido aos múltiplos transcritos associados a cada gene mRNA, a quantidade de pares aumentou após a transformação, com um total de **23.019** exemplos positivos, **7.855** exemplos negativos e **1.227.038** exemplos não-rotulados.

Em continuidade, foi utilizada a versão 2010 do software Miranda (ENRIGHT et al., 2003), para realizar alinhamento de sequência entre miRNAs e mRNA alvo. A configuração padrão foi usada na execução desse software. Para vários destes exemplos, não foram encontrados alinhamentos válidos e, portanto, o número de exemplos diminuiu. Ao final deste passo tínhamos um total de **7.100** exemplos positivos, **2.131** exemplos negativos e **490.201** exemplos não-rotulados. A tabela abaixo (Tabela 4.2) sumariza estes números.

Tabela 4.2: Quantidades de dados para cada passo da geração do *dataset*.

	Tarbase	Sequências	**Alinhamentos**
Positivos	5.619	23.019	**7.100**
Negativos	1.944	7.855	**2.131**
Não-rotulados	301.893	1.227.038	**490.201**
Total	309.456	1.257.912	**499.432**

Fonte: Autor

A partir desses dados, um *script* Perl foi utilizado para extrair as *features*, para

cada alinhamento válido. As *features* usadas nesse trabalho foram propostas por Mendoza et al. (2013) e são sumarizadas na Tabela 4.3.

Tabela 4.3: Relação de *features* usadas neste trabalho, baseadas no classificador RFMirTarget, proposto por Mendoza et al. (2013).

#	Nome da *feature*	#	Nome da *feature*
1	Alignment score	18	Position 10
2	Alignment length	19	Position 11
3	Minimum free energy of the alignment	20	Position 12
4	G:C's absolute frequency in the alignment	21	Position 13
5	A:U's absolute frequency in the alignment	22	Position 14
6	G:U's absolute frequency in the alignment	23	Position 15
7	Number of gaps in the alignment	24	Position 16
8	Number of mismatches in the alignment	25	Position 17
9	Position 1	26	Position 18
10	Position 2	27	Position 19
11	Position 3	28	Position 20
12	Position 4	29	Minimum free energy of the seed
13	Position 5	30	G:C's absolute frequency in the seed
14	Position 6	31	A:U's absolute frequency in the seed
15	Position 7	32	G:U's absolute frequency in the seed
16	Position 8	33	Number of gaps in the seed
17	Position 9	34	Number of mismatches in the seed

Fonte: Autor

4.3 Classificadores

Foram gerados doze métodos de classificação a partir de seis classificadores do pacote *scikit-learn* (PEDREGOSA et al., 2011) e dois métodos de *PU learning* (WRIGHT, 2017; MORDELET; VERT, 2014; KABOUTARI; BAGHERZADEH; KHERADMAND, 2014), descritos a seguir.

- *Baseline Random Forest*: Um classificador *Random Forest* treinado somente com os dados rotulados, consistindo de exemplos positivos e negativos. É utilizado como um exemplo dos resultados existentes obtidos com algoritmos supervisionados.
- *Random Forest*: Um classificador *Random Forest* treinado com os dados não-rotulados, juntamente com os dados rotulados, considerando os dados não-rotulados como negativos. Assim o conjunto de dados negativos é formado por exemplos negativos e não-rotulados.
- *Random Forest com Subsampling*: É similar ao método *Random Forest*, porém, utilizando a técnica de *subsampling*, para igualar as quantidades de dados positivos e negativos. Similar ao método anterior, os exemplos negativos incluem os dados

não-rotulados.

- *Baseline SVM, SVM com Subsampling*: A descrição é a mesma dos algoritmos *Baseline Random Forest* e *Random Forest com Subsampling*, porém, utilizando o classificador SVM.
- *BaggingPU Decision Trees*: O método *BaggingPU* utilizado é baseado no algoritmo definido por Mordelet and Vert (2014) e implementado por Wright (2017). Essa implementação é uma modificação do algoritmo *Bagging*, do pacote *scikit-learn*. Nesse caso foram utilizados 1000 classificadores de árvores de decisão.
- *BaggingPU SVM*: É o mesmo método de *BaggingPU Decision Trees*, no entanto, utiliza dez classificadores SVM. A quantidade menor de classificadores se dá, em razão do tempo maior de treino necessário para um algoritmo SVM, quando comparado ao algoritmo de árvore de decisão.
- *Two-Step*: O método utilizado foi implementado por Wright (2017), baseado em Kaboutari, Bagherzadeh and Kheradmand (2014). Esse método, como descrito em 2.3.2.1, é divido em dois passos. No primeiro passo, o método *Random Forest com Subsampling* foi utilizado como o classificador necessário para execução da técnica de geração dos *reliable negatives* e, no segundo passo, o classificador *Random Forest* foi utilizado para a classificação a partir dos dados positivos e *reliable negatives*, repetido por 3 iterações.
- *One-Class SVM*: Um classificador *One-Class SVM* treinado em modo *Outlier Detection*, ou seja, com todos os dados, mas sem informações de rótulo, por ser um classificador não-supervisionado. Este classificador não gera predições em formato de probabilidade como os demais classificadores, acima mencionados. Portanto, para gerar valores de probabilidade, um classificador supervisionado de Regressão Logística foi treinado com a saída do *One-Class SVM*, em formato de *score* dos dados de treino positivos e negativos, e usado para gerar os valores probabilísticos para os dados de teste.
- *Isolation Forest*: Um classificador *Isolation Forest* treinado com todos os dados, mas sem informações de rótulo (modo *Outlier Detection*), considerando tratar-se de um classificador não-supervisionado. Também utiliza o classificador de Regressão Logística para geração de valores probabilísticos.
- *Novelty One-Class SVM*: A mesma descrição do método One-Class SVM, mas treinado somente com os dados positivos rotulados (modo *Novelty Detection*).

- *Novelty Isolation Forest*: A mesma descrição do método Isolation Forest, mas treinado somente com os dados positivos rotulados (modo *Novelty Detection*).

As chamadas utilizadas para cada classificador, com seus respectivos parâmetros, podem ser vistas na Tabela 4.4. Os hiperparâmetros utilizados foram mantidos para todas as execuções dos métodos, a fim de ter uma comparação melhor das diferentes execuções. Assim, também, os hiperparâmetros utilizados para cada classificador foram mantidos, quando utilizados em diferentes métodos, para poder ter uma comparação dos métodos supervisionados em relação aos métodos de *PU learning*, sem influência dos hiperparâmetros.

Tabela 4.4: Chamadas dos classificadores do pacote *scikit-learn*.

Nome do Classificador	Chamada
Random Forest	RandomForestClassifier(n_estimators = 1000)
SVM	SVC(gamma='scale',probability = True))
Árvore de Decisão	DecisionTreeClassifier()
One-Class SVM	OneClassSVM(gamma='scale')
Isolation Forest	IsolationForest(n_estimators=1000, behaviour='new', contamination='auto')
Regressão Logística	LogisticRegression(solver='lbfgs', max_iter=1000))

Fonte: Autor

Para os classificadores SVM, *One-Class* SVM e Regressão Logística foi utilizado um algoritmo de normalização, por padronização (Subseção 2.3.4), transformando os dados para que cada *feature* tenha média 0 e variância unitária. Esses algoritmos dependem da normalização de todas as *features*, para que uma das *features* não afete a classificação mais do que as outras. Ao contrário dos algoritmos que usam árvores de decisão, pois estes não são afetados pela magnitude ou distribuição dos valores dos dados.

A Tabela 4.5 apresenta um resumo dos métodos utilizados.

4.4 Validação dos métodos

A validação dos métodos utilizados é feita com o algoritmo de validação cruzada *k-fold* estratificado com k=10. Métodos de validação cruzada são custosos em relação ao tempo de execução, mas tem um menor desperdício de dados para teste, além de uma menor variância nas medidas de desempenho, do que uma divisão simples de treino e teste dos dados. A motivação para o uso do algoritmo *k-fold* estratificado, ao invés do método *k-fold*, mais comumente utilizado, será explicada na Seção 5.1 do Capítulo de experimentos e resultados.

Tabela 4.5: Resumo dos métodos utilizados.

Nome do Método	Classificador	Extra
Baseline Random Forest	*Random Forest*	Treinado somente com os dados positivos e negativos.
Random Forest	*Random Forest*	Treinado utilizando os dados não-rotulados, juntamente aos dados negativos rotulados.
Random Forest com Subsampling	*Random Forest*	Igual ao método Random Forest, mas utilizando subsampling, para igualar as quantidades das classes positiva e negativa.
Baseline SVM	SVM	Treinado somente com os dados positivos e negativos.
SVM com Subsampling	SVM	Treinado utilizando os dados não-rotulados, juntamente aos dados negativos rotulados, mas utilizando subsampling, para igualar as quantidades das classes positiva e negativa.
BaggingPU Decision Trees	Árvore de Decisão	Método de *PU learning*. Utiliza 1000 classificadores de árvore de decisão.
BaggingPU SVM	SVM	Método de *PU learning*. Utiliza 10 classificadores SVM.
Two-Step		Método de *PU learning*. Utiliza o método Random Forest com Subsampling no primeiro passo, e o classificador *Random Forest* no segundo passo.
One-Class SVM	*One-Class* SVM	Método OCC. Treinado utilizando todos os dados, mas sem rótulos, por tratar-se de um algoritmo não-supervisionado.
Isolation Forest	*Isolation Forest*	Método OCC. Treinado utilizando todos os dados, mas sem rótulos, por tratar-se de um algoritmo não-supervisionado.
Novelty One-Class SVM	*One-Class* SVM	Método OCC. Treinado somente com os dados rotulados positivos.
Novelty Isolation Forest	*Isolation Forest*	Método OCC. Treinado somente com os dados rotulados positivos.

Fonte: Autor

O *dataset* deste trabalho possui as seguintes quantidades de dados para cada classe: 7.100 positivos, 2.131 negativos e 490.201 não-rotulados. Na maioria dos experimentos, este *dataset* foi modificado, às vezes com amostragem de alguma classe e às vezes com remoção de dados específicos. Para todos os casos essa modificação foi feita antes de executar o algoritmo de validação cruzada, a não ser, para os experimentos em que todos os negativos são removidos. Neste caso, os dados negativos foram removidos apenas do subconjunto de treino e não do subconjunto de teste.

Após a subdivisão dos *folds* da validação cruzada, foi feito também, a remoção dos dados não-rotulados presentes nos conjuntos de teste. Isto foi feito em razão de não sabermos a classe correta para estes dados e não podermos avaliar a classificação de dados que não possuem essa informação.

Em alguns experimentos foi feito uma amostragem dos dados não-rotulados, a fim de comparar o desempenho dos métodos, para diferentes quantidades de não-rotulados. Nesses casos, a amostragem é feita sem reposição e o tamanho da amostra é descrito nos próprios resultados.

4.5 Medidas de desempenho

Existem diversas medidas de desempenho de classificação possíveis para problemas de classificação binária. Todas essas medidas são baseadas na matriz de confusão e seus valores, que compreendem os verdadeiros positivos, verdadeiros negativos, falsos

positivos e falsos negativos.

A métrica mais simples baseada na matriz de confusão é a acurácia, que é definida como a quantidade total de dados classificados corretamente dividido pelo total de dados testados. Esta medida, no entanto, não é efetiva para dados de teste desbalanceados, isto é, uma classe com quantidade maiores do que outras, visto que pode fornecer visão enviesada do desempenho. Como nesse trabalho os conjuntos de teste possuem a mesma distribuição de classes do *dataset* como um todo, sendo que os positivos correspondem a 77% dos dados rotulados, a medida de acurácia teria um valor de 77%, para um classificador que classificasse todos os dados como positivos. Por causa disso, outras medidas mais robustas ao desbalanceamento foram necessárias.

Para a predição de alvos de microRNA, o interesse maior está na classe positiva ao invés da classe negativa. A classe positiva indica alvos reais, e o objetivo da predição de alvos é avaliar pares de miRNA e mRNA para os quais não se tem informações se o miRNA possui alvo no mRNA, e tentar encontrar os pares que têm a maior chance de possuírem alvos reais, para posteriormente serem verificados experimentalmente.

A curva ROC, explicada em 2.3.6.2, é a medida perfeita para esse caso, por mostrar o *trade-off* entre as taxas de verdadeiros positivos e falsos positivos ao variar a quantidade de dados classificados como positivos. Por isso, as medidas utilizadas nesse trabalho são a curva ROC e a área sob a curva ROC (score AUC).

4.6 Análise Estatística

A significância estatística da diferença de desempenho entre os métodos foi verificada com o teste estatístico *Student's t-test*, adotando um nível de significância de 95%, o que corresponde a um *p-value* igual ou menor a 0,05.

5 EXPERIMENTOS E RESULTADOS

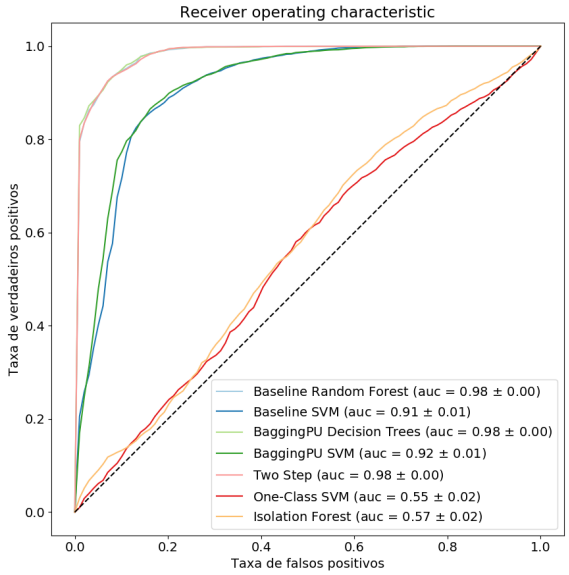
Utilizando as métricas descritas na Seção 4.5, os métodos propostos foram testados com várias configurações diferentes dos dados. As descrições dessas configurações, dos experimentos e análise dos resultados seguem nas próximas seções.

5.1 Correlação dos dados e exemplos duplicados

Os dados do Tarbase contêm o identificador do gene mRNA, mas não especificam o transcrito. Como os transcritos são variações de um mesmo gene, eles têm sequências de nucleotídeos similares e, após passar pelo Miranda (Seção 4.1.1), para a geração dos alinhamentos possíveis, os alinhamentos gerados podem ser similares ou até mesmo duplicados.

Nas primeiras execuções dos experimentos, utilizou-se um algoritmo de validação cruzada que embaralha os dados antes de fazer a divisão. Os resultados em termos de score AUC eram praticamente perfeitos para os algoritmos que utilizavam árvores de decisão (Figura 5.1). Estes resultados, obviamente, não eram representativos de um desempenho real desses algoritmos.

Figura 5.1: Curva ROC. Resultado com embaralhamento dos dados antes da divisão dos subconjuntos.



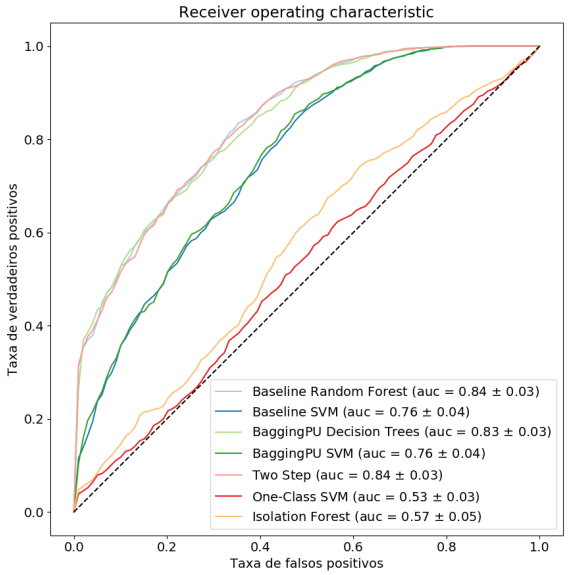
Fonte: Autor

A hipótese inicial, com esses resultados tão altos, era que os transcritos estavam gerando alinhamentos muito similares e, portanto, dados com *features* de valores simila-

res. Para resolver este problema foi aplicado a versão de validação cruzada, sem o embaralhamento inicial dos dados. Como no nosso *dataset* os dados provindos de diferentes transcritos de um mesmo par miRNA e mRNA aparecem em sequência, a separação dos k-subconjuntos sem reordenar os dados aleatoriamente, mantinha esses dados dos diferentes transcritos dentro de um mesmo subconjunto. Nesse caso, os dados de diferentes transcritos, para o mesmo par miRNA e mRNA, não apareceriam no conjunto de treino e de teste ao mesmo tempo.

Na forma como estão dispostos os dados no *dataset* seguindo a sequência de dados positivos, negativos e, por último, não-rotulados, ao aplicarmos a validação cruzada sem reordenamento dos dados, serão gerados subconjuntos com todos os dados de uma mesma classe. Para resolver este problema, foi utilizado a validação cruzada *k-fold* estratificada que efetua a divisão dos subconjuntos, mantendo as proporções das diferentes classes.

Figura 5.2: Curva ROC. Resultado sem embaralhamento dos dados.

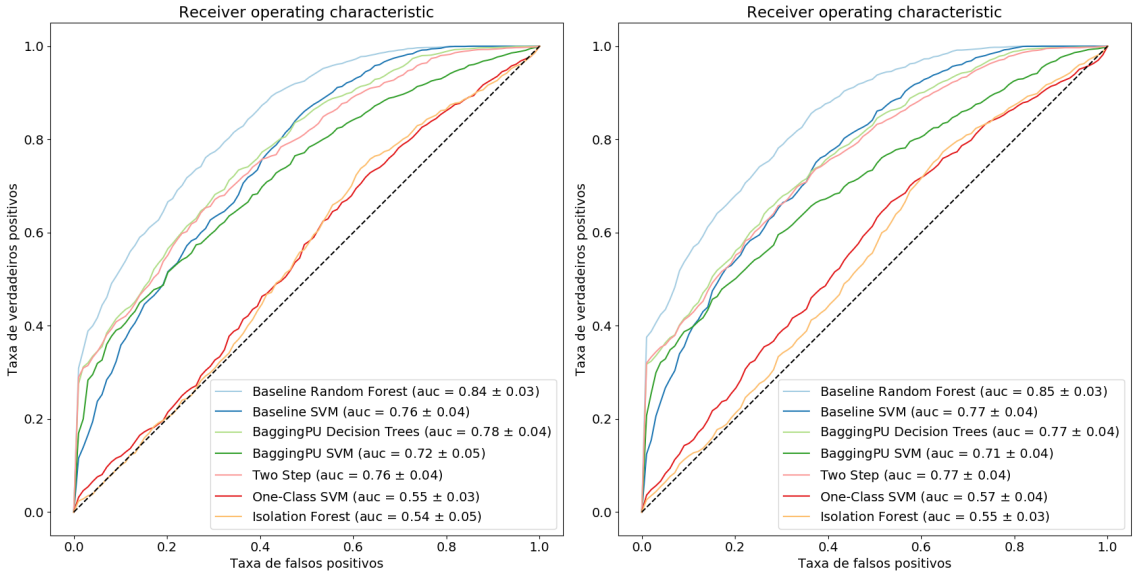


Fonte: Autor

Os resultados obtidos sem o reordenamento aleatório dos dados (Figura 5.2) não foram perfeitos, como verificado anteriormente, quando reordenados aleatoriamente (Figura 5.1). Considerando resolvido o problema de correlação dos dados, todos os experimentos abaixo demonstrados foram executados utilizando validação cruzada *k-fold* estratificada sem embaralhamento dos dados. Ocorre que, após a execução de todos os experimentos, verificou-se que o *dataset* apresentava não só dados similares, conforme suposto inicialmente, mas também, dados duplicados. Os dados duplicados não foram removidos do *dataset* para a execução dos experimentos, considerando a necessidade de mais tempo para re-executar todos os experimentos, e dado o avançado estágio do desen-

volvimento. No entanto, os dados duplicados, assim como os dados similares, aparecem em sequência no *dataset*, por serem provindos dos transcritos de um mesmo mRNA. Ao utilizar o algoritmo de validação cruzada sem reordenamento dos dados, esses dados duplicados não aparecem nos conjuntos de treino e teste ao mesmo tempo e, portanto, não devem influenciar os resultados de maneira significativa. Para fins de verificação dessa hipótese, um exemplo de execução sem os dados duplicados é demonstrado na Figura 5.3, juntamente com os resultados de uma execução comparativa com os duplicados. Podemos observar que o desempenho foi pouco impactado pela presença de dados duplicados. Assim, enfatizamos que este fator não deve alterar qualitativamente os resultados obtidos.

Figura 5.3: Curva ROC. resultados de execuções com duplicados a esquerda e sem duplicados a direita, e 10.000 dados não-rotulados.



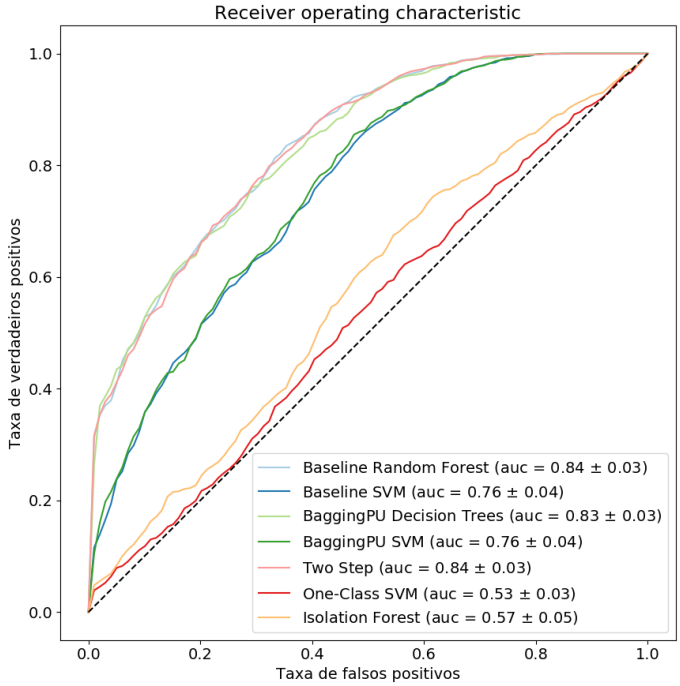
Fonte: Autor

5.2 Algoritmos supervisionados, *PU learning* e *One-class Classification*

A hipótese inicial desse trabalho era a de que algoritmos que aprendem a partir de dados rotulados apenas da classe positiva, sem precisar de dados rotulados da classe negativa, tenham desempenho melhor do que algoritmos supervisionados, por não serem afetados pelo desbalanceamento das classes positiva e negativa.

Além do desbalanceamento das classes, os dados negativos não são necessariamente representativos da distribuição real da classe negativa, devido a sua pequena quantidade. Isso também poderia afetar a classificação de algoritmos supervisionados, porém, não afetaria os algoritmos *PU Learning* e OCC. Assim, neste primeiro experimento

Figura 5.4: Curva ROC. resultados de execução sem usar dados não-rotulados.



Fonte: Autor

traçou-se uma comparação entre os métodos adotados.

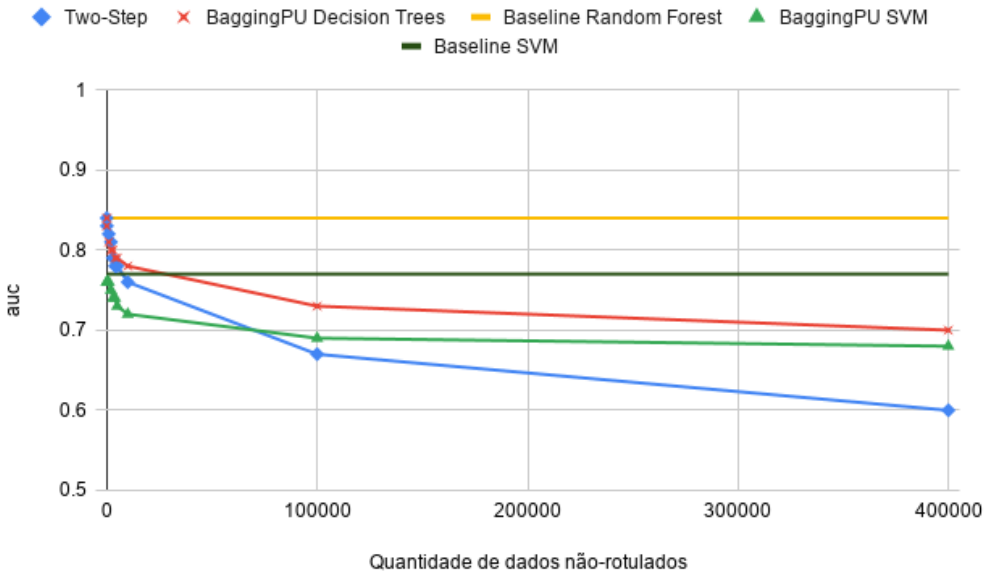
Em relação à Figura 5.4, verifica-se que os resultados dos métodos de *PU Learning*, sem usar os dados não-rotulados, são equivalentes ao resultados dos métodos supervisionados.

Em relação à Figura 5.5, verifica-se que os resultados com *PU learning* mostraram-se piores ao aumentar a quantidade de dados não-rotulados, e que os dados não-rotulados, além de não ajudar, atrapalham o desempenho destes métodos. Estes resultados sugerem que para os dados não-rotulados disponíveis e as *features* usadas, os dados não-rotulados não estão contribuindo para refinar a fronteira de decisão como deveriam, conforme explicado na Seção 2.3.2. Isto pode ser decorrente da estratégia adotada em nossa metodologia para geração de dados não-rotulados, de forma que os resultados poderiam ser diferentes para conjuntos distintos de dados não-rotulados.

Conforme Figura 5.6, com relação a treinamento com todos os dados não-rotulados, verifica-se que os resultados dos algoritmos de *PU learning* mostram-se piores em relação aos algoritmos supervisionados com um valor estatisticamente significativo.

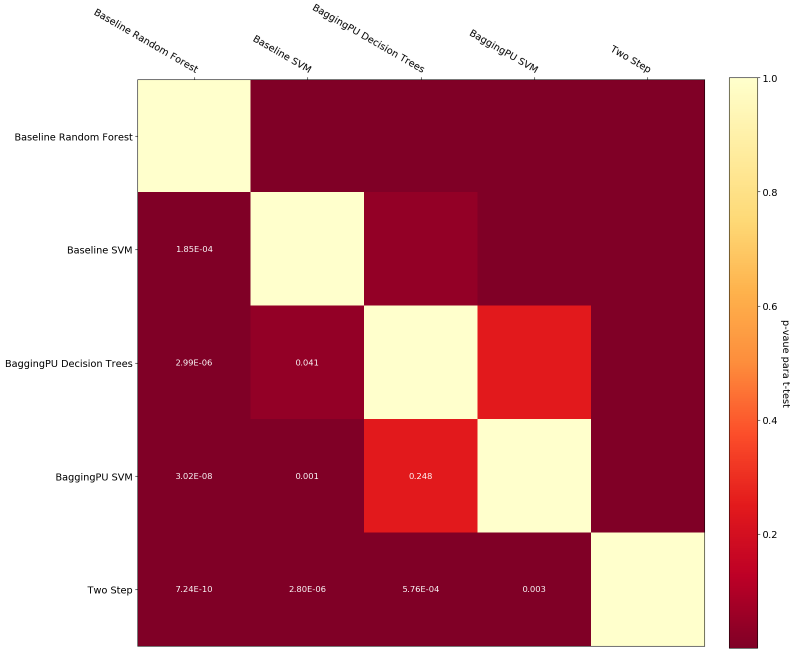
Conforme a Figura 5.7, os resultados dos métodos OCC mostraram-se bastante inferiores em relação aos métodos *PU learning* e supervisionados. Observa-se ainda, que os resultados dos métodos OCC sobem e descem, porém, não demonstram ser diretamente afetados pela quantidade de dados não-rotulados. Os resultados dos algoritmos OCC em

Figura 5.5: Score AUC dos métodos de *PU learning* para execuções com diferentes quantidades de dados não-rotulados. O AUC dos algoritmos Baseline é mostrado como referência, ele não muda para as diferentes quantidades, porque não utilizada dados não-rotulados.



Fonte: Autor

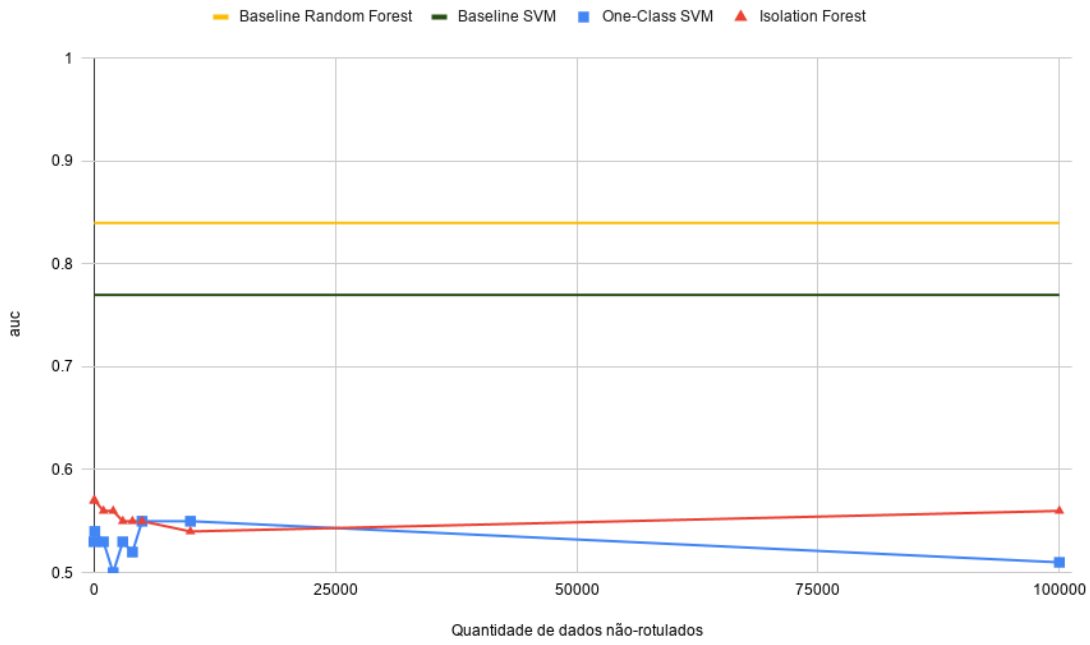
Figura 5.6: Matriz de *p-values* para execução utilizando todos os dados não-rotulados.



Fonte: Autor

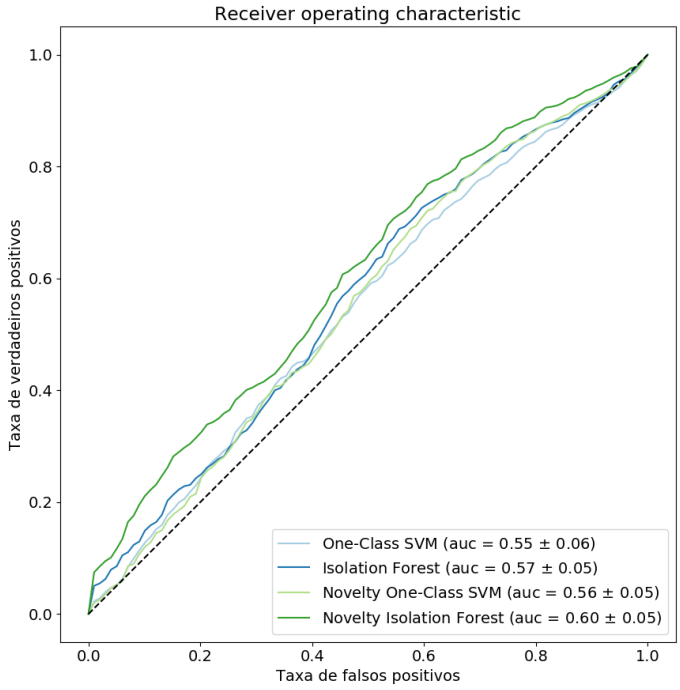
modo *Novelty Detection* (Figura 5.8) foram um pouco melhores do que os resultados em modo *Outlier Detection* mas não com um valor estatisticamente significativo.

Figura 5.7: Score AUC para execuções dos algoritmos OCC com diferentes quantidades de dados não-rotulados.



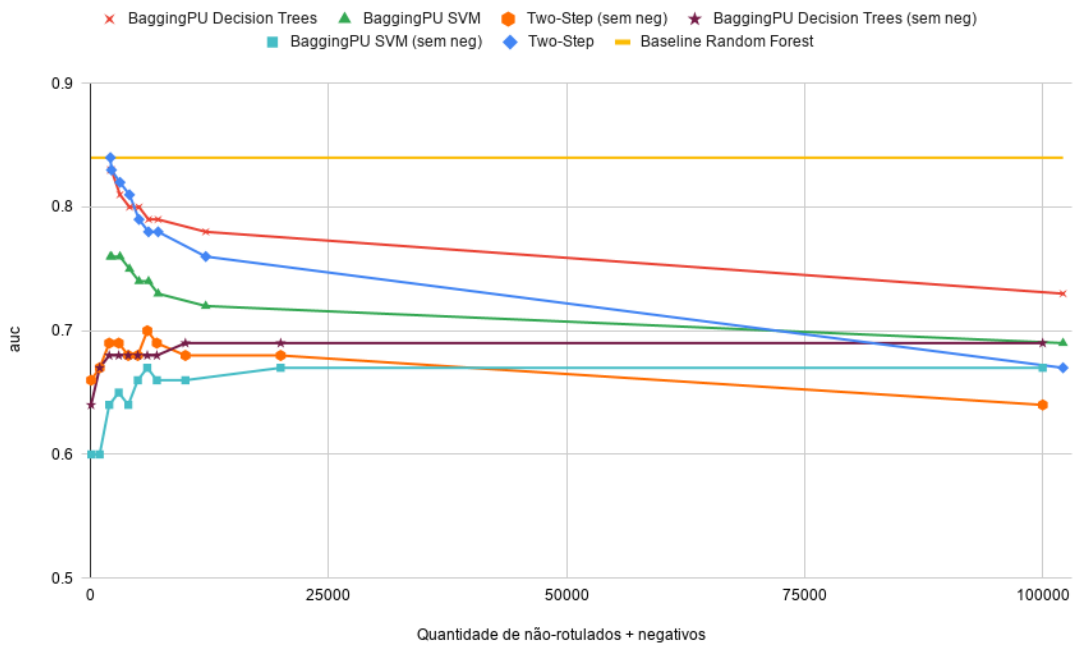
Fonte: Autor

Figura 5.8: Score AUC para execuções dos algoritmos OCC em modo *Outlier Detection* sem dados não-rotulados, e modo *Novelty Detection*.



Fonte: Autor

Figura 5.9: Score AUC para diferentes quantidades de dados não-rotulados. Os resultados com o rótulo "(sem neg)" são os resultados removendo os dados negativos.



Fonte: Autor

5.3 Treinamento removendo os dados negativos

Os resultados a seguir demonstrados, foram verificados em treinamentos sem dados negativos, para testar o desempenho dos algoritmos usando somente os dados não-rotulados, além dos dados rotulados positivos. Essa é a maneira em que os algoritmos de *PU Learning* foram pensados para serem usados, no caso em que não existem dados rotulados negativos, apenas positivos.

Em relação à Figura 5.9, verifica-se que a quantidade de dados não-rotulados melhora o desempenho até chegar próximo da quantidade de dados positivos, quando o desempenho para de aumentar. Os piores desempenhos observados com poucos dados não-rotulados decorrem do fato de que os algoritmos *PU learning* não foram feitos para serem usados com quantidades pequenas de dados não-rotulados. O método *PU-Bagging* faz amostragem da classe majoritária. Sendo majoritária a classe positiva, a amostra é feita a partir da referida classe e, portanto, ocorre a perda de dados rotulados no treinamento. O resultado, nesse caso, é um desempenho pior se a quantidade de dados não-rotulados for menor que a quantidade de dados positivos.

O método Two-Step utilizado, também realiza amostragem da classe majoritária, porque usa o algoritmo Random Forest com Subsampling no primeiro passo do método.

Portanto, apresenta o mesmo problema do método *PU-Bagging*, quando a classe positiva for majoritária.

Em relação aos algoritmos de *PU learning*, os resultados sem exemplos negativos são piores do que com negativos, porém melhores do que os algoritmos OCC. Isso demonstra que os dados negativos são um contraponto melhor aos positivos, e ajudam mais a refinar a classificação com relação aos dados não-rotulados. Por outro lado, os dados não-rotulados contém informações úteis para a classificação e apresentam melhores resultados que um treinamento somente com os dados positivos, como ocorre com os algoritmos OCC.

Com base nos resultados acima, a hipótese do trabalho de que um treinamento com dados não-rotulados utilizando aprendizado semi-supervisionado seria melhor do que um treinamento com algoritmos supervisionados, levando-se em conta os problemas dos dados negativos disponíveis descritos anteriormente, foi rejeitada.

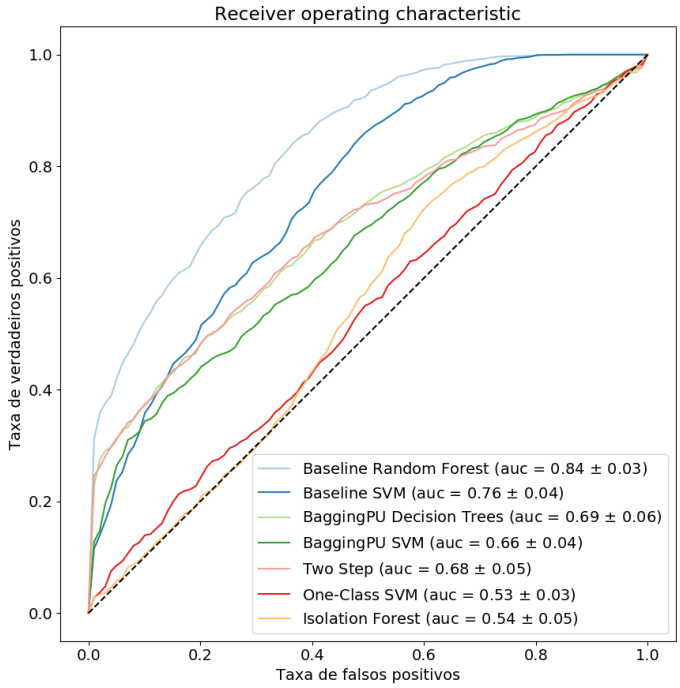
Os resultados mostram que os dados não-rotulados contém informações úteis, mas de pior qualidade que os dados negativos, não adicionando nenhuma informação relevante para determinar a fronteira de decisão neste domínio. Esses dados não-rotulados, quando usados juntamente com os dados negativos, acabam apenas dissolvendo os dados negativos no meio dos dados não-rotulados e diminuindo com isso o desempenho.

Como explicado no início desta seção, o treinamento sem dados negativos é o treinamento da maneira que *PU learning* foi pensado para ser usado. Se fosse o caso de não se ter nenhum dado negativo, *PU learning* seria a escolha certa, já que é significativamente melhor do que OCC, conforme demonstrado nas Figuras 5.10 e 5.11. No entanto, considerando que temos dados rotulados negativos, mesmo que em pouca quantidade e com problemas descritos anteriormente, os algoritmos supervisionados, se não significativamente melhores (Figura 5.12), são pelo menos, tão bons quanto os algoritmos de *PU learning*, e não necessitam gerar um *dataset* de dados não-rotulados.

5.4 Algoritmos supervisionados treinados com dados não-rotulados

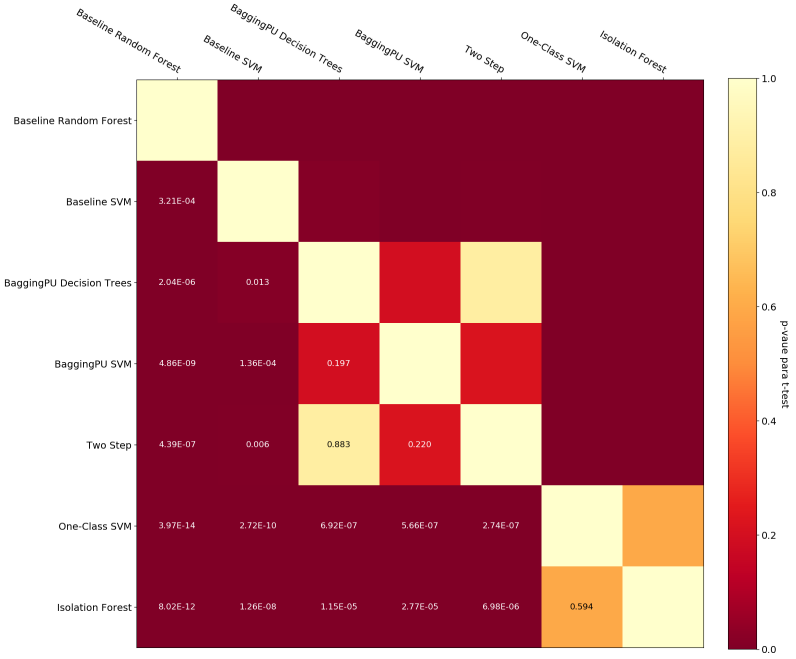
Essa seção contém os resultados das execuções dos métodos Random Forest, Random Forest com Subsampling e SVM com subsampling, realizando o treinamento com os dados não-rotulados considerados como exemplos negativos. Esses métodos são treinados, portanto, com os mesmos dados que os métodos *PU learning*, ao contrário dos métodos Baseline que são treinados somente com os dados rotulados. O objetivo deste

Figura 5.10: Curva ROC para treinamento sem dados negativos e 10.000 dados não-rotulados. Os métodos Baseline foram mantidos como comparação e foram treinados normalmente, com dados negativos.



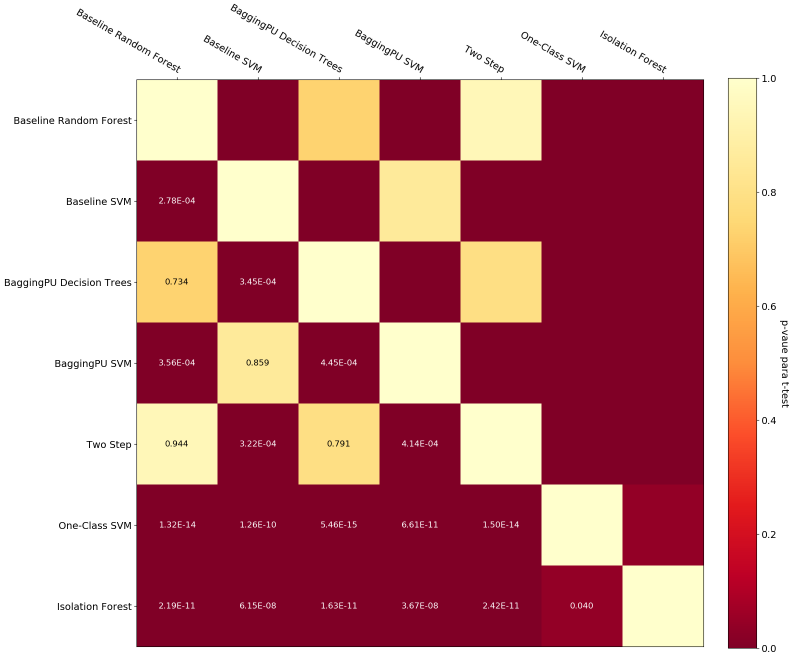
Fonte: Autor

Figura 5.11: Matriz de p-values, para execução sem negativos e 10.000 não-rotulados. Os métodos Baseline foram mantidos como comparação e foram treinados normalmente, com dados negativos.



Fonte: Autor

Figura 5.12: Matriz de p-values, para execução sem utilizar dados não-rotulados.



Fonte: Autor

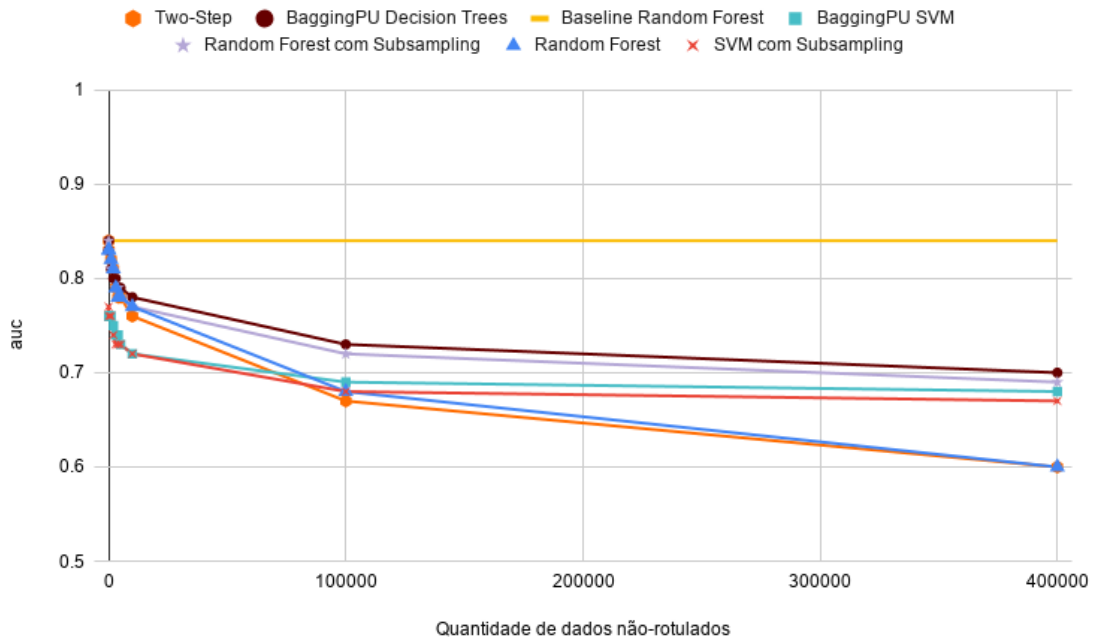
experimento é comparar o desempenho do *PU learning* com algoritmos supervisionados treinados com os mesmos dados.

Um método SVM sem usar Subsampling não foi utilizado, porque o classificador SVM não escala bem com o tamanho da entrada, por ter complexidade quadrática. O tempo de execução seria infactível usando grandes quantidades de dados não-rotulados.

Os resultados dos treinamentos, com e sem dados negativos (Figuras 5.13 e 5.14), de Random Forest com Subsampling são similares aos de PU Bagging Decision Trees, os resultados de Random Forest são similares aos de Two-Step e os resultados de SVM com Subsampling são similares aos de PU Bagging SVM. Estes resultados mostram, novamente, que os algoritmos de *PU learning* não estão conseguindo tirar nenhuma informação extra dos dados não-rotulados com relação aos dados negativos.

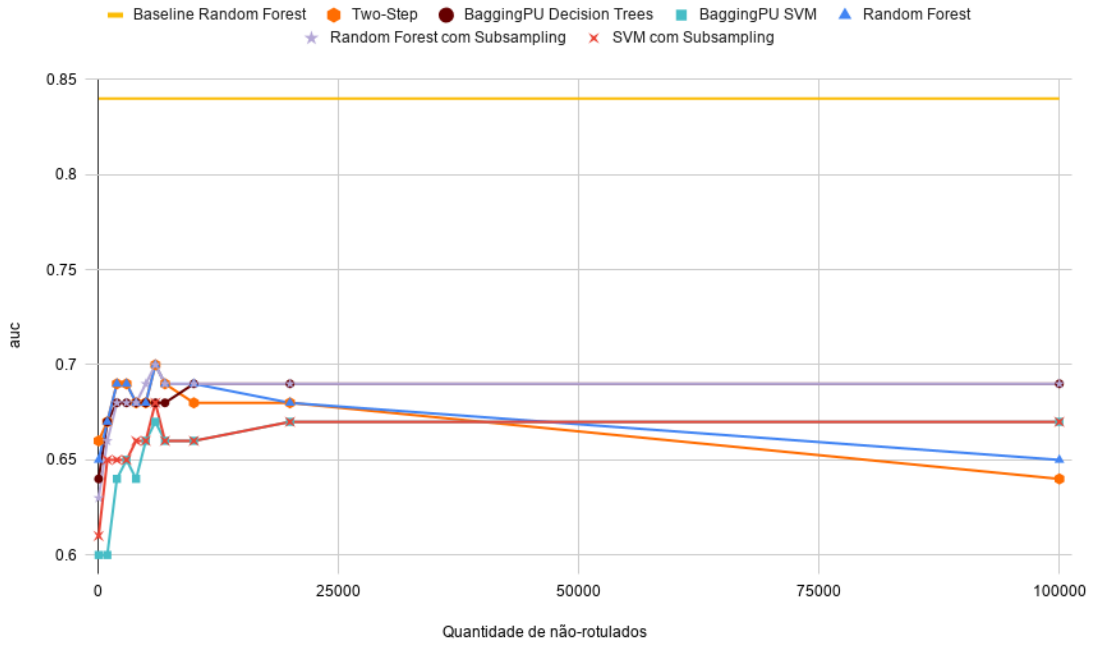
Não é possível afirmar que um algoritmo supervisionado, treinado com os dados não-rotulados como negativos, terá sempre um desempenho tão bom quanto os métodos de *PU learning* utilizados. No entanto, para esse dataset e para as features utilizadas, os algoritmos de *PU learning* não tiveram nenhum ganho significativo de desempenho, mesmo considerando a hipótese de nenhum dado rotulado negativo e apenas dados não-rotulados, como foi mostrado nas Figuras 5.14 e 5.15.

Figura 5.13: Score AUC para diferentes quantidades de dados não-rotulados.



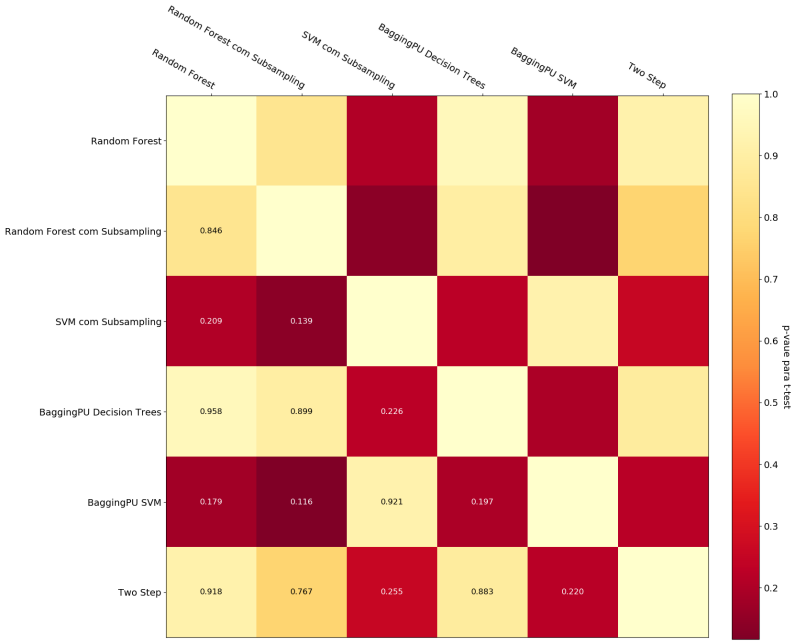
Fonte: Autor

Figura 5.14: Score AUC para diferentes quantidades de dados não-rotulados. treinamento sem negativos.



Fonte: Autor

Figura 5.15: Matriz de p-values, para treinamento sem negativos e 10.000 exemplos não-rotulados.



Fonte: Autor

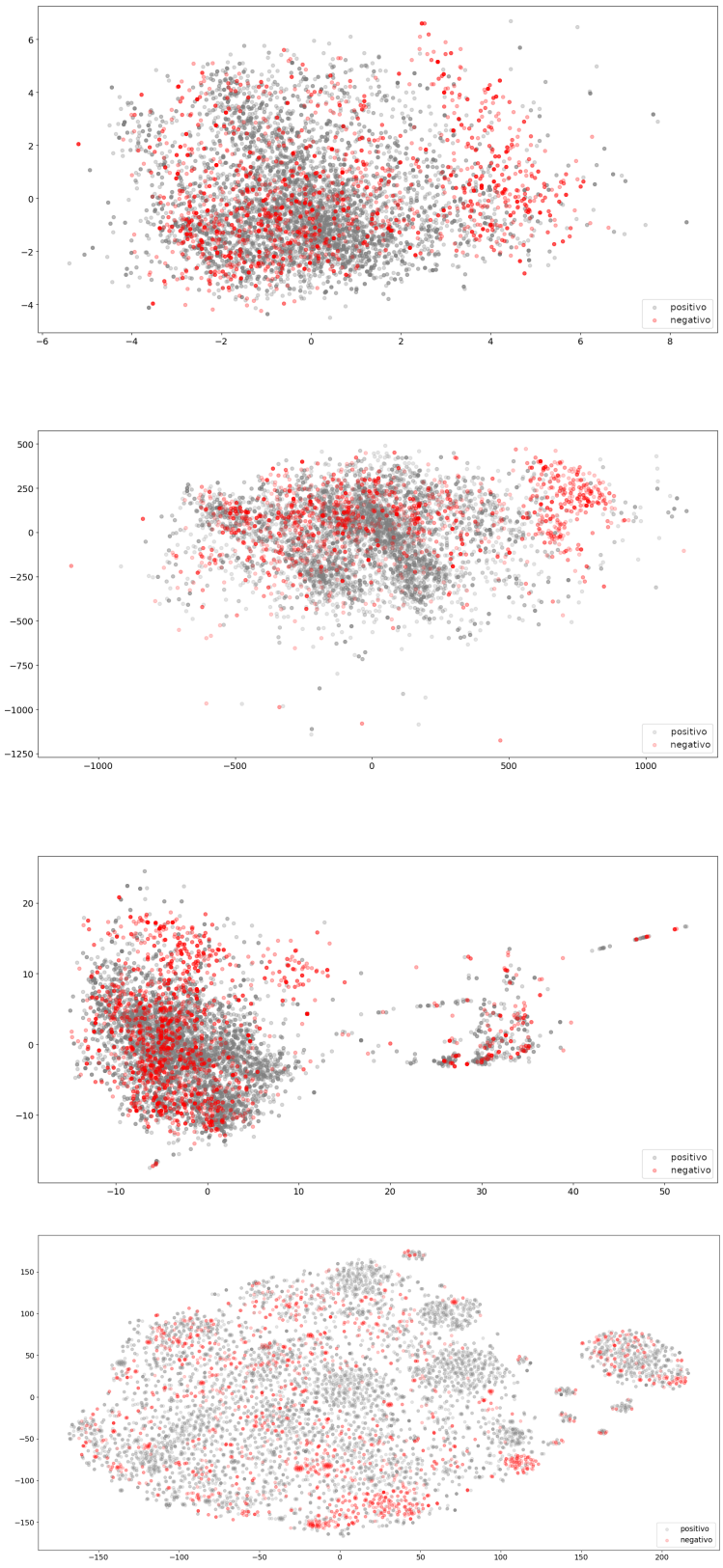
5.5 Visualização dos dados

Utilizando algoritmos de redução de dimensionalidade descritos no referencial teórico, para reduzir o espaço do problema para duas dimensões, podemos visualizar as distribuições dos dados. As imagens geradas pelos três primeiros algoritmos (Figura 5.16) não têm *clusters* claros de positivos ou negativos, a não ser por um *cluster* de negativos que aparece em todas as imagens e fica mais evidente na imagem do NCA. Este *cluster* está localizado na área entre 800 e 1200 no eixo X e -150 e 500 no eixo Y. O algoritmo T-SNE tem uma separação melhor das classes do que os outros algoritmos, mas não mostra o *cluster* de negativos tão bem quanto o NCA.

Nas Figuras 5.17 e 5.18 pode ser visto que os dados não-rotulados estão espalhados por toda a área abrangida pelos dados positivos e negativos. Isto faz sentido, pois os dados não-rotulados são uma mistura de dados positivos e negativos.

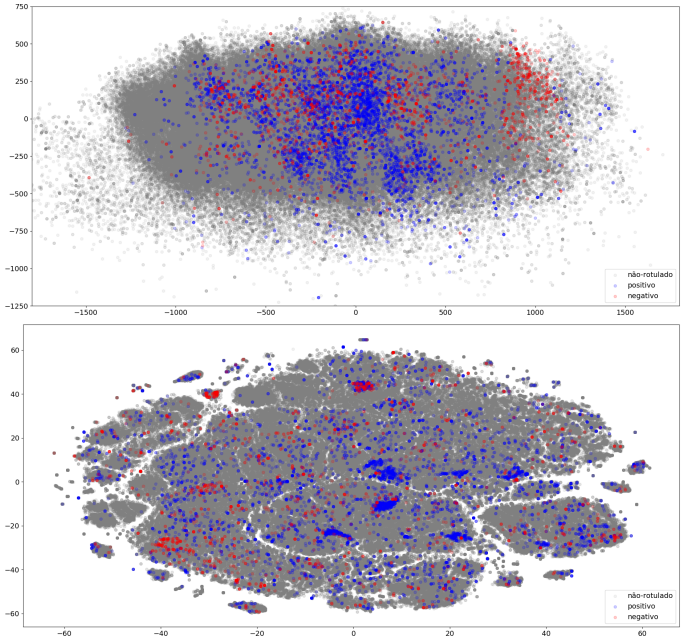
Não é possível fazer afirmações fortes com relação a esses gráficos gerados, dado que eles são apenas visualizações 2D de dados que contém 35 *features* e, portanto, 35 dimensões. No entanto, os algoritmos de redução de dimensionalidade não são simplesmente projeções aleatórias dos dados. Esses algoritmos tentam manter os *clusters* e a estrutura geral dos dados, e gerar visualizações com separações claras entre as classes. No entanto, para os dados não-rotulados não é possível visualizar *clusters* de dados bem

Figura 5.16: Visualização 2D de dados positivos e negativos para os algoritmos de redução de dimensionalidade, em ordem: PCA, NCA, isomap e T-SNE. Para o algoritmo T-SNE foram utilizados os parâmetros: taxa de aprendizado-200, número de iterações-5000 e Perplexidade-50



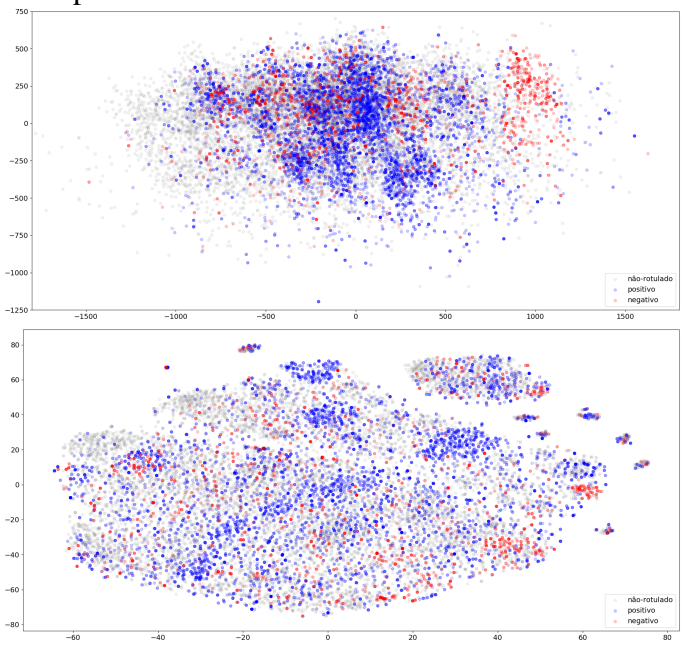
Fonte: Autor

Figura 5.17: Visualização com todos os dados não-rotulados usando os algoritmos NCA e T-SNE respectivamente. Parâmetros do T-SNE: taxa de aprendizado-200, número de iterações-2000 e Perplexidade-50.



Fonte: Autor

Figura 5.18: Visualização com 10.000 exemplos não-rotulados usando os algoritmos NCA e T-SNE respectivamente. Parâmetros do T-SNE: taxa de aprendizado-50, número de iterações-2000 e Perplexidade-50.



Fonte: Autor

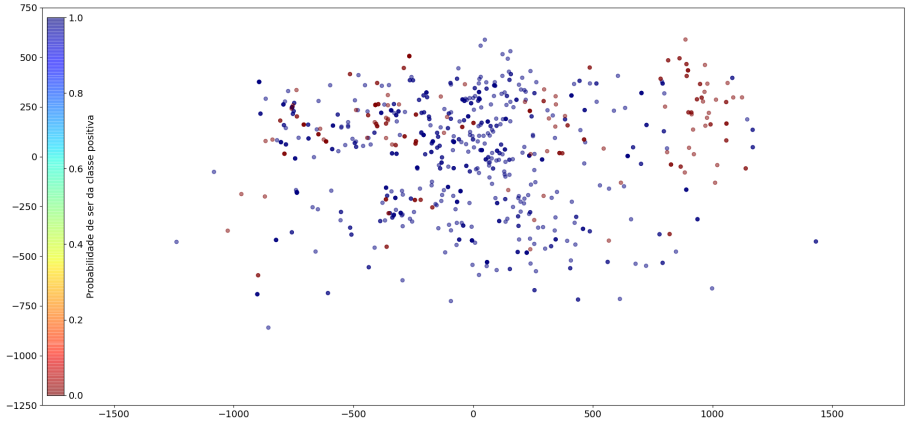
definidos. Nas imagem do algoritmo NCA, os dados não-rotulados têm uma distribuição uniforme. Já nas imagens do algoritmo T-SNE, existem *clusters*, mas quase todos eles contém dados rotulados positivos e negativos. Para ajudar a separar melhor as classes, é necessário que um *cluster* de dados não rotulados contenha dados rotulados de uma única classe, já que assim, os algoritmos semi-supervisionados podem usar a estrutura do cluster para refinar a fronteira de decisão (seção 2.3.2).

O fato de os dados não-rotulados não conterem *clusters* bem definidos e serem basicamente uniformes, levando-se em conta que, algoritmos semi-supervisionados usam os *clusters* e a estrutura geral dos dados não-rotulados para refinar a classificação dos dados rotulados, pode explicar o porquê de os métodos de *PU learning* testados não conseguirem gerar nenhum ganho de desempenho de classificação com relação aos algoritmos supervisionados.

5.6 Visualização das probabilidades geradas pelos métodos

Usando os algoritmos de redução de dimensionalidade, também foi possível gerar visualizações das classificações geradas pelos métodos em forma de probabilidade. As visualizações das Figuras 5.19, 5.20 e 5.21 são do conjunto de teste de um dos folds da validação cruzada. Os valores de probabilidade apresentados são valores para cada instância avaliada, e indicam a probabilidade da instância ser da classe positiva. A primeira Figura (5.19) contém as probabilidades esperadas para os dados de teste utilizando as informações de rótulo positivo e negativo dos dados.

Figura 5.19: Visualização dos dados de teste positivos e negativos.

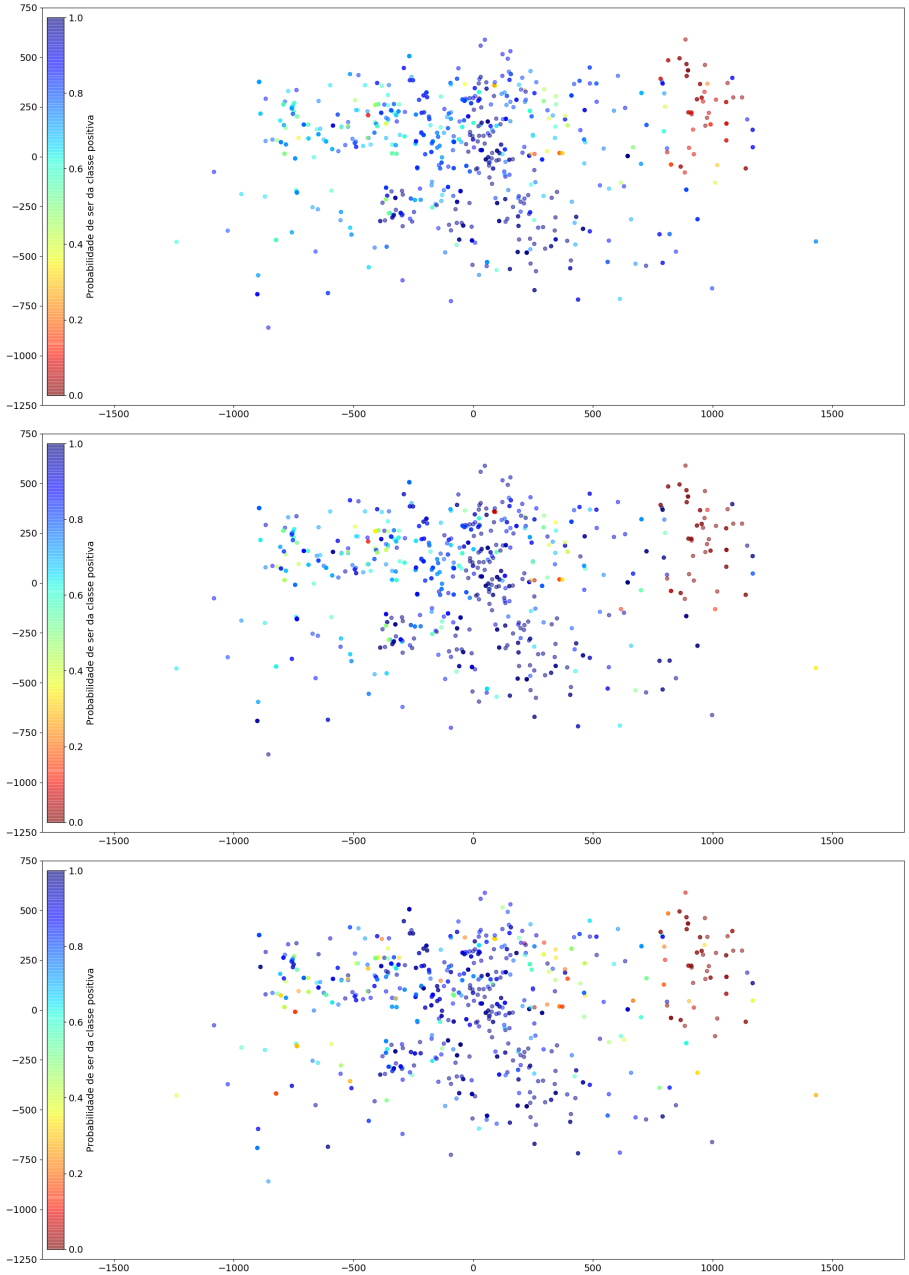


Fonte: Autor

As Figuras 5.20 e 5.21 mostram que todos os algoritmos supervisionados e de *PU*

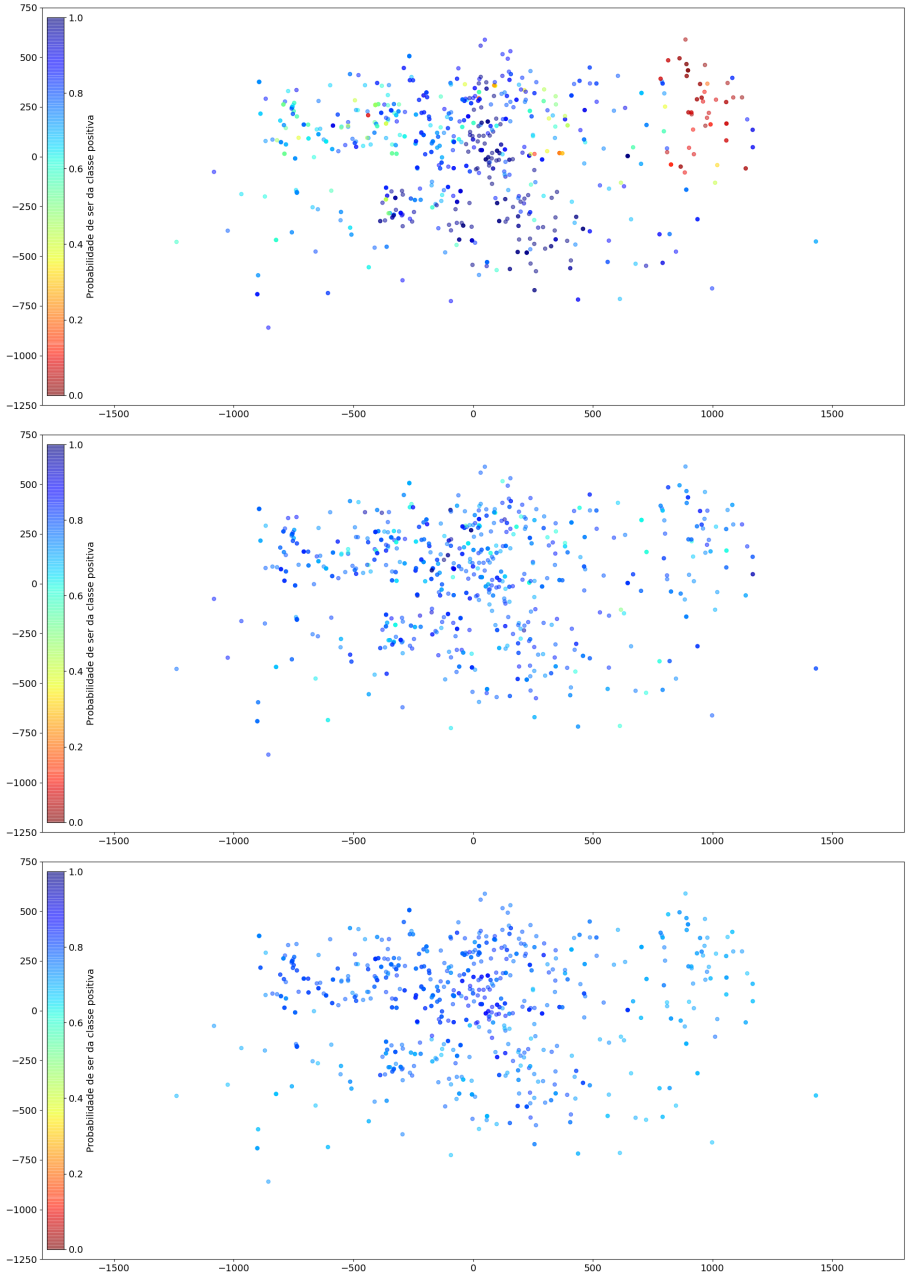
learning aprendem bem a classificar os dados negativos do cluster mencionado na Seção 5.5. Para o restante dos dados, no entanto, os algoritmos tem dificuldade em diferenciar positivos de negativos. Os algoritmos OCC geram probabilidades altas para todas as instâncias e não conseguem diferenciar positivos de negativos.

Figura 5.20: Probabilidades geradas para, em ordem: Baseline Random Forest, PU bagging Decision Trees e PU Bagging SVM.



Fonte: Autor

Figura 5.21: Probabilidades geradas para, em ordem: Two-Step, one-class SVM e isolation forest.

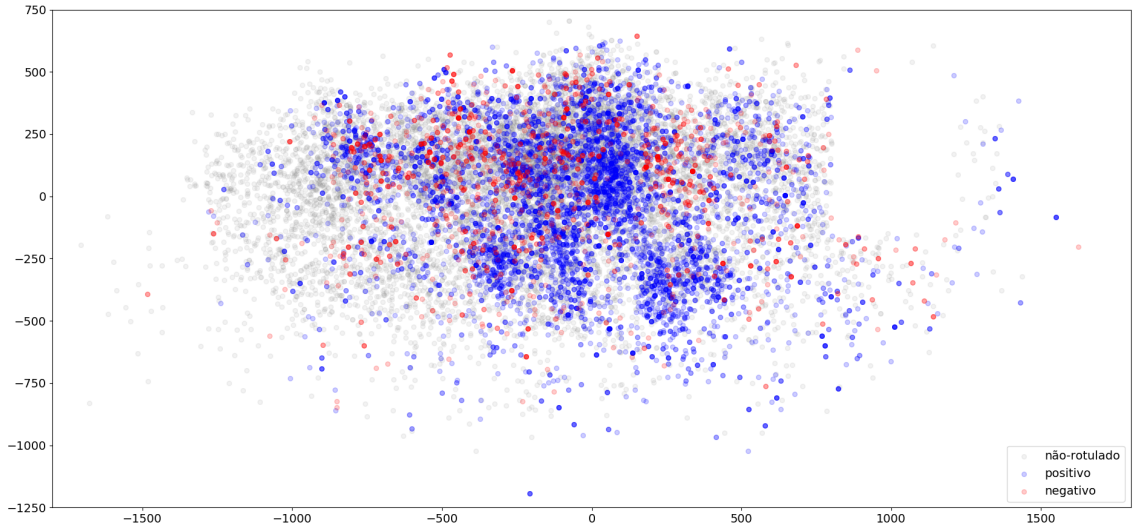


Fonte: Autor

5.7 Treinamento removendo o *cluster* de negativos

Baseado nas visualizações das probabilidades geradas para os algoritmos e no fato de que os algoritmos supervisionados e de *PU learning* pareciam classificar muito bem o *cluster* de negativos, mas não o restante dos dados, foi feita uma execução dos algoritmos removendo todos os dados da área desse *cluster* de 800 a 1200 no eixo X e -150 a 500 no eixo Y. Visualização do *dadaset* modificado na Figura 5.22. O objetivo desse experimento

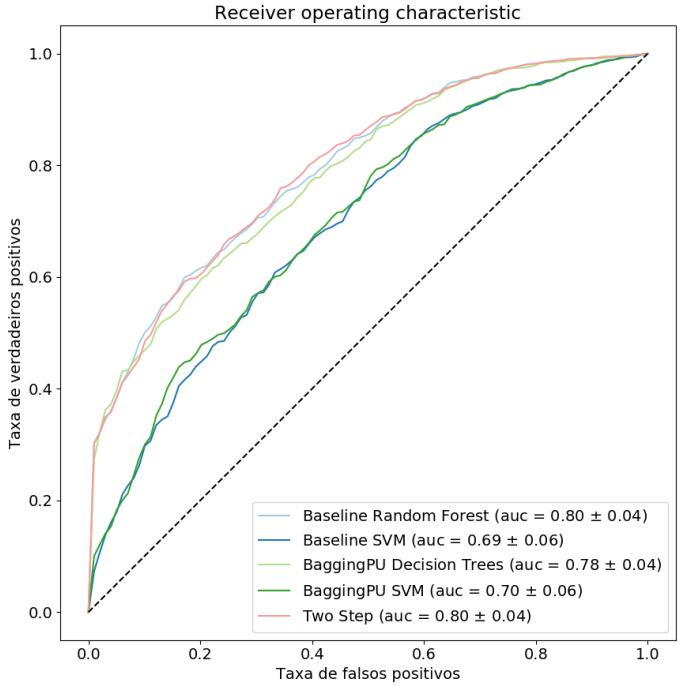
Figura 5.22: Visualização do *dataset* sem o *cluster* de exemplos negativos, para uma execução com 10.000 dados não-rotulados.



Fonte: Autor

é testar o desempenho desses algoritmos para o resto dos dados (excluindo o *cluster*), e testar o quanto o desempenho cairia com relação ao experimento com todos os positivos e negativos.

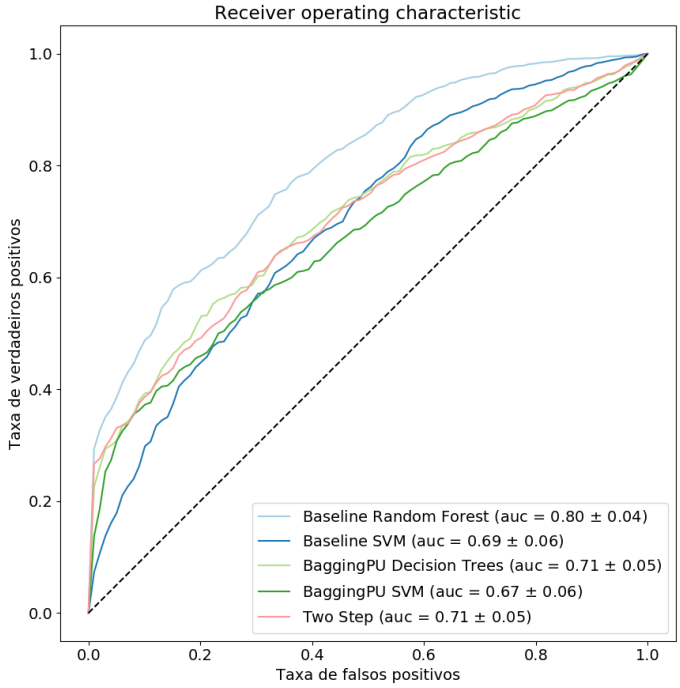
Figura 5.23: Curva ROC para uma execução do *dataset* sem o *cluster* de exemplos negativos, sem nenhum dado não-rotulado.



Fonte: Autor

O desempenho dos métodos (Figuras 5.23 e 5.24) caiu significativamente quando comparado ao desempenho com todo o *dataset* de positivos e negativos, e o desvio padrão

Figura 5.24: Curva ROC para uma execução do *dataset* sem o *cluster* de exemplos negativos, com 10.000 dados não-rotulados.



Fonte: Autor

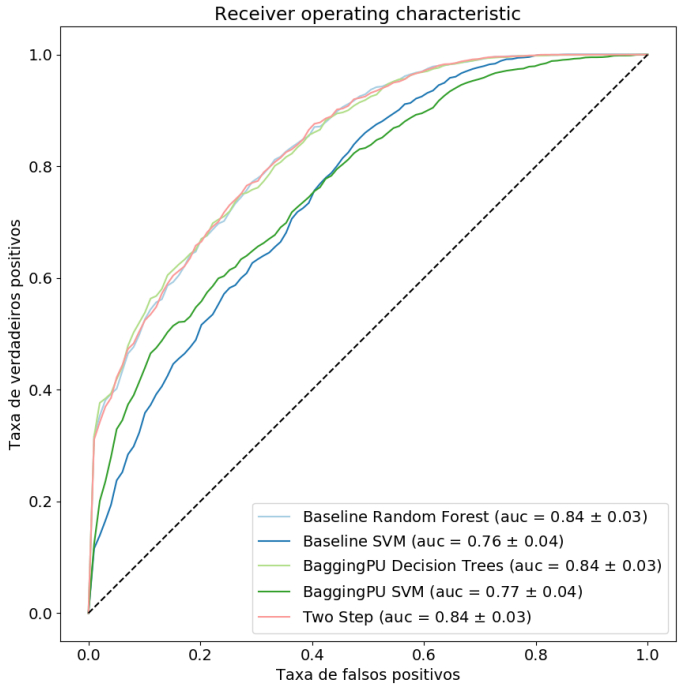
do AUC também aumentou para quase todos os algoritmos. Com relação ao AUC dos experimentos sem os dados negativos (Seção 5.3), o desempenho dos algoritmos nesse experimento, sem o *cluster*, ainda foi significativamente melhor. Isso mostra que o ganho de desempenho dos dados negativos em cima dos dados não-rotulados não se deve só ao *cluster* de negativos. A qualidade dos dados negativos como um contraponto aos positivos ainda prevalece, mesmo sem o *cluster*.

5.8 Treinamento com inversão de exemplos positivos e negativos no conjunto de treino

Os dados não-rotulados foram gerados à partir dos dados experimentais da base de dados Tarbase, utilizando os dados considerados funcionais (positivos), mas com evidência fraca (Seção 4.1).

Uma hipótese possível é a de que os dados não-rotulados tenham uma semelhança maior aos dados positivos e, seria possível que o treinamento de *PU learning* tivesse melhores resultados, trocando os rótulos das classes positivas e negativas para treino. A troca de rótulo das classes significa, para o treinamento, que os dados negativos são considerados como rotulados e os dados não-rotulados e positivos como não-rotulados.

Figura 5.25: Curva ROC para treinamento com as classes positiva e negativa trocadas e nenhum dado não-rotulado.



Fonte: Autor

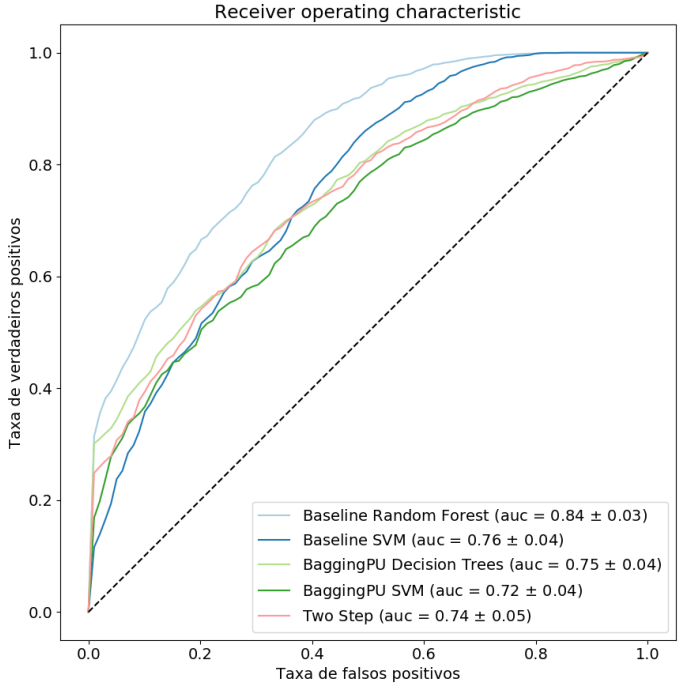
A suposição nesse caso é a de que os dados não-rotulados junto dos positivos são um contraponto melhor aos negativos do que os negativos e não-rotulados são aos positivos.

A probabilidade gerada pelos métodos com as classes trocadas será a probabilidade da instância ser da classe negativa e não da classe positiva, como nos experimentos anteriores e, no entanto, os dados de teste mantém as classes originais sem serem trocadas. Dessa forma, a probabilidade utilizada nos teste é 1 menos a probabilidade gerada pelos classificadores, o que troca a predição de probabilidade da classe negativa para a predição da classe positiva.

Os resultados sem utilizar dados não-rotulados (Figura 5.25) foram similares ao treinamento sem trocar as classes. Isto mostra que a implementação foi feita corretamente e os resultados são válidos mesmo após fazer todas essas operações sobre os dados.

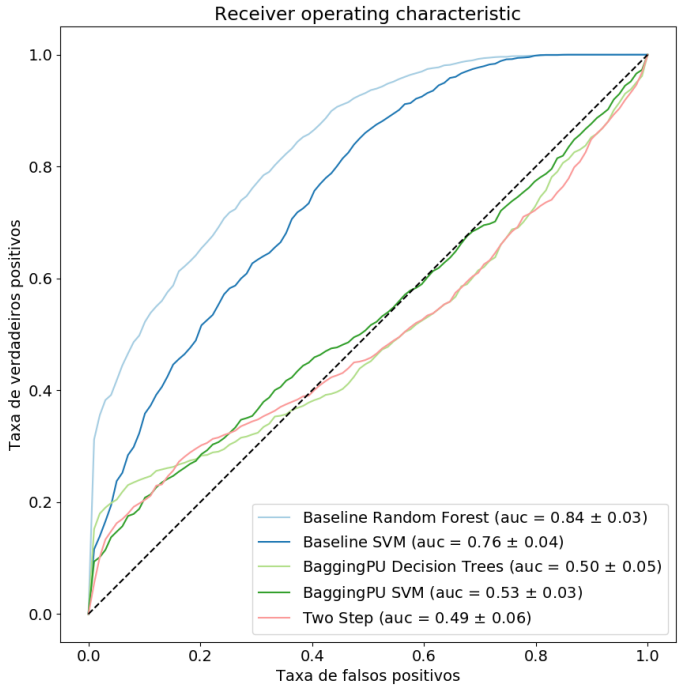
Os resultados com 1000 e 10.000 dados não-rotulados (Figuras 5.26 e 5.27) mostram uma queda acentuada de desempenho, muito maior do que a queda demonstrada na Figura 5.5 que contém resultados sem trocar as classes. Com 10.000 dados não-rotulados o AUC de todos métodos ficou próximo de 0.5, que é o valor de AUC para um classificador randômico, sem nenhum aprendizado.

Figura 5.26: Curva ROC para treinamento com as classes positiva e negativa trocadas e 1.000 dados não-rotulados.



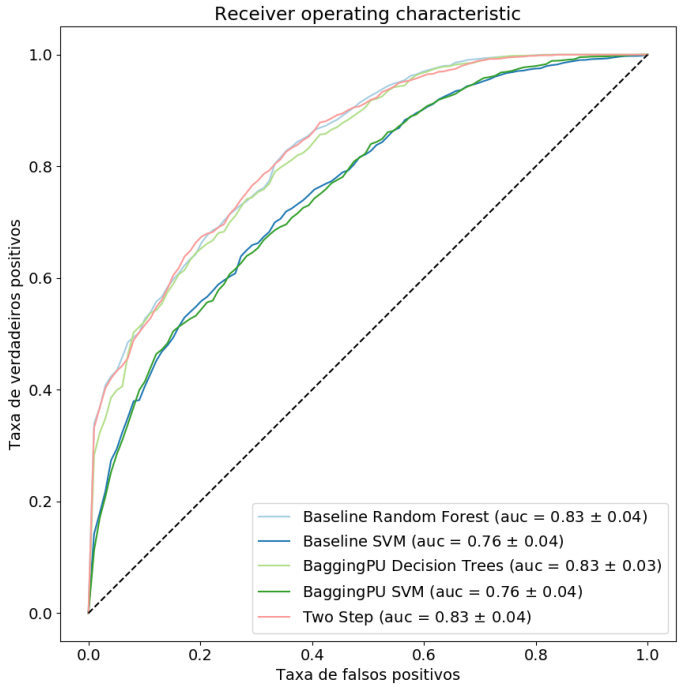
Fonte: Autor

Figura 5.27: Curva ROC para treinamento com as classes positiva e negativa trocadas e 10.000 dados não-rotulados.



Fonte: Autor

Figura 5.28: Curva ROC para treinamento igualando as classes positiva e negativa, e nenhum dado não-rotulado.



Fonte: Autor

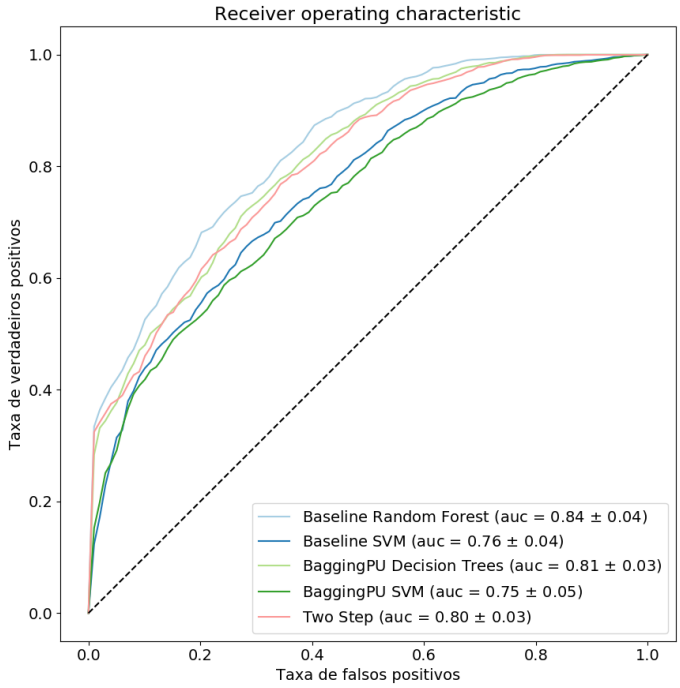
5.9 Igualando as quantidade de dados positivos e negativos

A comparação dos resultados da Seção anterior (5.8) com os resultados descritos nas seções anteriores sem trocar as classes não é, necessariamente justa se considerarmos que a quantidade de dados positivos é muito maior do que a quantidade de dados negativos. O *dataset* contém 7100 positivos e 2131 negativos. Ao trocar as classes, foi gerado um conjunto de dados rotulados com menos da metade da quantidade de dados rotulados do experimento sem trocar as classes. Considerando que os dados rotulados influenciam mais no desempenho dos classificadores do que os dados não-rotulados, o desempenho inferior verificado ao trocar as classes seria de se esperar e, não indica, necessariamente, que o desempenho seria pior do que execuções sem trocar as classes se as quantidades de rotulados fossem as mesmas.

Para fazer então, uma comparação mais justa, foi feito um experimento sem trocar as classes e mantendo para os dados positivos apenas uma amostra de 2131 instâncias, igualando a quantidade de dados negativos. Portanto, ao treinar os algoritmos de *PU learning*, a quantidade de dados rotulados é a mesma dos experimentos trocando as classes positiva e negativa.

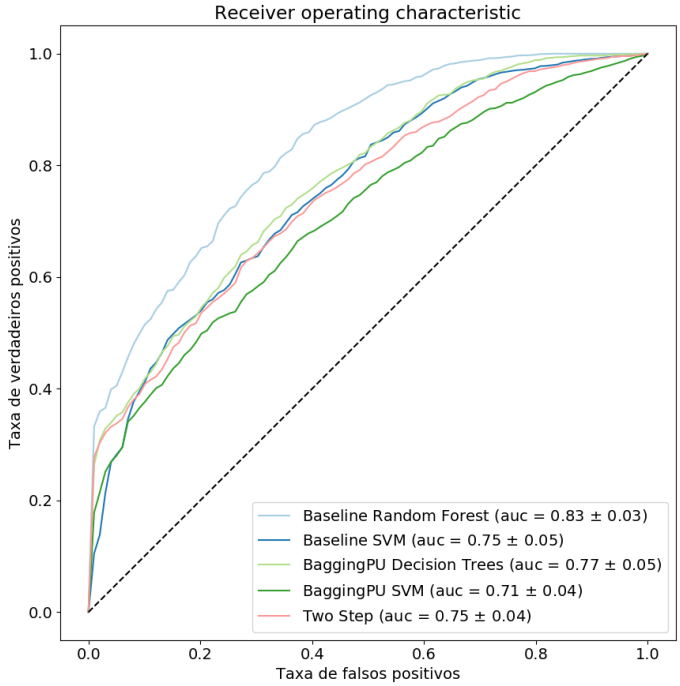
Como pode ser visto nas Figuras 5.28, 5.29 e 5.30, os resultados sem trocar as

Figura 5.29: Curva ROC, para treinamento igualando as classes positiva e negativa, e 1.000 dados não-rotulados.



Fonte: Autor

Figura 5.30: Curva ROC, para treinamento igualando as classes positiva e negativa, e 10.000 dados não-rotulados.



Fonte: Autor

classes continuam definitivamente melhores do que os resultados trocando as classes. Também é interessante apontar que esses resultados, utilizando somente 2131 positivos,

não são significativamente piores em relação ao resultados usando todos os positivos mostrados na Seção 5.2.

6 CONCLUSÃO

No presente trabalho foi comparado o desempenho de classificação de algoritmos supervisionados, *PU Learning* e OCC para o problema de predição de alvos de miRNAs. Baseado nos resultados demonstrados no Capítulo anterior, pode-se concluir que, para o *dataset* e as *features* utilizados, em relação aos algoritmos supervisionados, não há nenhum ganho de desempenho em se usar métodos de *PU Learning*. Além disso, observou-se que os métodos OCC são muito inferiores, tanto em relação a algoritmos supervisionados, quanto a métodos de *PU Learning*.

Os algoritmos OCC tiveram desempenho muito fraco, levemente acima do desempenho de um algoritmo aleatório, sem aprendizado. Os algoritmos *PU Learning* tiveram desempenho equivalente aos algoritmos supervisionados, quando não eram adicionados os dados não-rotulados. Quando adicionou-se todos os dados não-rotulados, o desempenho foi significativamente inferior e, no geral, quanto mais dados não-rotulados foram utilizados, piores foram os resultados dos algoritmos de *PU Learning*.

Nas seções 5.3 até 5.9 foi estudado as características dos dados rotulados e não-rotulados, e os motivos do desempenho inferior dos algoritmos de *PU Learning* ao utilizar dados não-rotulados:

- No treinamento sem dados negativos (Seção 5.3), verificou-se que os dados não-rotulados são inferiores aos dados negativos, quando utilizados como contraponto aos dados positivos.
- Nos experimentos com os algoritmos supervisionados treinados com dados não-rotulados (Seção 5.4), verificou-se que os métodos de *PU Learning* não estão extraindo nenhuma informação extra dos dados não-rotulados, quando comparados a algoritmos supervisionados treinados com os mesmos dados.
- A visualização dos dados (Seção 5.5) demonstrou uma distribuição uniforme dos dados não-rotulados, o que pode atrapalhar o desempenho de algoritmos semi-supervisionados como o *PU learning*.
- Nas seções 5.5 e 5.6 apontou-se o *cluster* de exemplos negativos. Os algoritmos aprendem a diferenciar bem esse *cluster*, mas não o restante dos dados.
- Os treinamentos sem o *cluster* de negativos (Seção 5.7) mostraram que os dados negativos continuam gerando um desempenho melhor do que os dados não-rotulados, mesmo sem o *cluster*.

- Nas seções 5.8 e 5.9, os treinamentos com inversão de exemplos positivos e negativos nos conjuntos de treino mostraram que os dados não-rotulados, também, não são um bom contraponto aos dados negativos e não servem como substitutos aos exemplos positivos rotulados.

Para trabalhos futuros seria interessante utilizar dados de uma base de dados de alvos de miRNAs independente do Tarbase, considerando estes dados como o conjunto de teste, para se ter uma medida melhor do desempenho real desses algoritmos. É possível que os algoritmos de *PU Learning* estejam tendo algum ganho com relação ao algoritmos supervisionados, mas esse ganho não apareça nos testes efetuados. Os métodos de *PU Learning* podem estar aprendendo, através dos dados não-rotulados, sobre dados positivos e negativos que não aparecem na distribuição de dados rotulados do *dataset* utilizado. Nesse caso, o desempenho melhor do *PU Learning*, para esses dados, não apareceria nas medidas de desempenho. Adicionalmente, seria possível explorar diferentes estratégias para gerar dados não-rotulados, tendo em vista que as decisões adotadas no presente trabalho para definir um exemplo como dado não-rotulado podem exercer certa influência nos resultados dos algoritmos semi-supervisionados.

Outra possibilidade seria usar *features* diferentes das utilizadas. As *features* utilizadas são baseadas no conhecimento atual dos elementos que influenciam um alvo de miRNA ser funcional ou não-funcional e, seguem a mesma linha que outros trabalhos da área. No entanto, outras *features* são possíveis, dado que as regras que regem as interações de mRNAs e miRNAs não são completamente entendidas e novas regras continuam sendo encontradas. Também seria possível utilizar métodos de extração de *features*, para encontrar novas *features* sem precisar se basear no conhecimento atual da literatura.

Uma última possibilidade seria estudar o que os métodos estão aprendendo a partir dos dados, tanto para os dados rotulados, quanto para os não-rotulados. As visualizações das probabilidades geradas neste trabalho mostram alguns detalhes interessantes com relação ao aprendizado e servem para comparar as classificações geradas pelos diferentes métodos. No entanto, existe espaço para um estudo mais aprofundado utilizando técnicas de interpretação do aprendizado de máquina, e uma análise melhor da diferença dos dados rotulados e não-rotulados.

REFERÊNCIAS

ALVES, C. A. B. Regulação da osteoglicina pelo microRNA-22 durante a miogênese. Universidade Estadual Paulista (UNESP), 2016.

BARTEL, D. P. MicroRNAs: target recognition and regulatory functions. **cell**, Elsevier, v. 136, n. 2, p. 215–233, 2009.

BISHOP, C. M. **Pattern recognition and machine learning**. [S.l.]: Springer Science+ Business Media, 2006.

BREIMAN, L. Random forests. **Machine learning**, Springer, v. 45, n. 1, p. 5–32, 2001.

BUSHATI, N.; COHEN, S. M. microRNA functions. **Annu. Rev. Cell Dev. Biol.**, Annual Reviews, v. 23, p. 175–205, 2007.

CAMPBELL, M.; FARREL, S. Bioquímica. **ARTMED-Sul: Porto Alegre**, 2001.

CHAPELLE, O.; SCHLKOPF, B.; ZIEN, A. **Semi-Supervised Learning**. 1st. ed. [S.l.]: The MIT Press, 2010. ISBN 0262514125, 9780262514125.

CHEN, X.; YAN, G.-Y. Semi-supervised learning for potential human microRNA-disease associations inference. **Scientific reports**, Nature Publishing Group, v. 4, p. 5501, 2014.

CHENG, S. et al. MiRTDL: a deep learning approach for miRNA target prediction. **IEEE/ACM transactions on computational biology and bioinformatics**, IEEE, v. 13, n. 6, p. 1161–1169, 2015.

DING, J.; LI, X.; HU, H. TarPmiR: a new approach for microRNA target site prediction. **Bioinformatics**, Oxford University Press, v. 32, n. 18, p. 2768–2775, 2016.

ELKAN, C.; NOTO, K. Learning classifiers from only positive and unlabeled data. In: **Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining**. New York, NY, USA: ACM, 2008. (KDD '08), p. 213–220. ISBN 978-1-60558-193-4. Available from Internet: <http://doi.acm.org/10.1145/1401890.1401920>.

ENRIGHT, A. J. et al. MicroRNA targets in Drosophila. **Genome biology**, BioMed Central, v. 5, n. 1, p. R1, 2003.

GOLDBERGER, J. et al. Neighbourhood components analysis. In: **Advances in neural information processing systems**. [S.l.: s.n.], 2005. p. 513–520.

GRIFFITHS-JONES, S. et al. miRBase: microRNA sequences, targets and gene nomenclature. **Nucleic acids research**, Oxford University Press, v. 34, n. suppl_1, p. D140–D144, 2006.

GUO, X. et al. On the class imbalance problem. In: IEEE. **2008 Fourth international conference on natural computation**. [S.l.], 2008. v. 4, p. 192–201.

JOHN, B. et al. Human microRNA targets. **PLoS biology**, Public Library of Science, v. 2, n. 11, p. e363, 2004.

JOLLIFFE, I. **Principal component analysis**. [S.l.]: Springer, 2011.

KABOUTARI, A.; BAGHERZADEH, J.; KHERADMAND, F. An evaluation of two-step techniques for positive-unlabeled learning in text classification. **Int. J. Comput. Appl. Technol. Res**, v. 3, p. 592–594, 2014.

KHAN, S. S.; MADDEN, M. G. A survey of recent trends in one class classification. In: SPRINGER. **Irish conference on artificial intelligence and cognitive science**. [S.l.], 2009. p. 188–197.

KOZOMARA, A.; GRIFFITHS-JONES, S. miRBase: annotating high confidence microRNAs using deep sequencing data. **Nucleic acids research**, Oxford University Press, v. 42, n. D1, p. D68–D73, 2013.

LEE, B. et al. deepTarget: end-to-end learning framework for microRNA target prediction using deep recurrent neural networks. In: ACM. **Proceedings of the 7th ACM International Conference on Bioinformatics, Computational Biology, and Health Informatics**. [S.l.], 2016. p. 434–442.

LIU, F. T.; TING, K. M.; ZHOU, Z.-H. Isolation forest. In: IEEE. **2008 Eighth IEEE International Conference on Data Mining**. [S.l.], 2008. p. 413–422.

MAATEN, L. v. d.; HINTON, G. Visualizing data using t-sne. **Journal of machine learning research**, v. 9, n. Nov, p. 2579–2605, 2008.

MARSLAND, S. **Machine learning: an algorithmic perspective**. [S.l.]: Chapman and Hall/CRC, 2014.

MENDOZA, M. R. et al. RFMirTarget: predicting human microRNA target genes with a random forest classifier. **PloS one**, Public Library of Science, v. 8, n. 7, p. e70153, 2013.

MITCHELL, T. M. et al. Machine learning. 1997. **Burr Ridge, IL: McGraw Hill**, v. 45, n. 37, p. 870–877, 1997.

MOITA, G. M. **Combining performance and diversity measures for optimizing classification ensembles via a genetic algorithm in the miRNA-target prediction problem**. 2018. Bachelor Thesis, UFRGS.

MONARD, M. C.; BARANAUSKAS, J. A. Conceitos sobre aprendizado de máquina. In: **Sistemas Inteligentes Fundamentos e Aplicações**. 1. ed. Barueri-SP: Manole Ltda, 2003. p. 89–114. ISBN 85-204-168.

MORDELET, F.; VERT, J.-P. A bagging svm to learn from positive and unlabeled examples. **Pattern Recognition Letters**, v. 37, p. 201 – 209, 2014. ISSN 0167-8655. Partially Supervised Learning for Pattern Recognition. Available from Internet: <http://www.sciencedirect.com/science/article/pii/S0167865513002432>.

PEDREGOSA, F. et al. Scikit-learn: Machine learning in python. **Journal of machine learning research**, v. 12, n. Oct, p. 2825–2830, 2011.

PEDREGOSA, F. et al. **Cross-validation Visualization**. 2019. Acessado em 18/10/2019. Available from Internet: <https://scikit-learn.org/stable/auto_examples/model_selection/plot_cv_indices.html>.

PEDREGOSA, F. et al. **standardization**. 2019. Acessado em 18/10/2019. Available from Internet: <https://scikit-learn.org/stable/modules/preprocessing.html#standardization-or-mean-removal-and-variance-scaling>.

PIO, G. et al. Integrating microRNA target predictions for the discovery of gene regulatory networks: a semi-supervised ensemble learning approach. **BMC bioinformatics**, BioMed Central, v. 15, n. 1, p. S4, 2014.

SILVA, V. D.; TENENBAUM, J. B. Global versus local methods in nonlinear dimensionality reduction. In: **Advances in neural information processing systems**. [S.l.: s.n.], 2003. p. 721–728.

SMEDLEY, D. et al. The biomart community portal: an innovative alternative to large, centralized data repositories. **Nucleic acids research**, Oxford University Press, v. 43, n. W1, p. W589–W598, 2015.

VAPNIK, V. N. The nature of statistical learning theory. Springer-Verlag, 1995.

VLACHOS, I. S. et al. DIANA-TarBase v7. 0: indexing more than half a million experimentally supported miRNA:mRNA interactions. **Nucleic acids research**, Oxford University Press, v. 43, n. D1, p. D153–D159, 2014.

VOET, D.; VOET, J. G. **Bioquímica**. [S.l.]: Artmed Editora, 2013.

WANG, Y. et al. Mechanism of microRNA-target interaction: molecular dynamics simulations and thermodynamics analysis. **PLoS computational biology**, Public Library of Science, v. 6, n. 7, p. e1000866, 2010.

WEN, M. et al. DeepMirTar: a deep-learning approach for predicting human miRNA targets. **Bioinformatics**, Oxford University Press, v. 34, n. 22, p. 3781–3787, 2018.

WRIGHT, R. **Positive-unlabeled learning**. 2017. Acessado em 18/10/2019. Available from Internet: <https://roywrightme.wordpress.com/2017/11/16/positive-unlabeled-learning/>.

YOUSEF, M.; ALLMER, J.; KHALIFA, W. Feature selection for microRNA target prediction comparison of one-class feature selection methodologies. SciTePress, 2016.

YOUSEF, M. et al. Learning from positive examples when the negative class is undetermined-microRNA gene identification. **Algorithms for Molecular Biology**, BioMed Central, v. 3, n. 1, p. 2, 2008.

YOUSEF, M.; NAJAMI, N.; KHALIFAV, W. A comparison study between one-class and two-class machine learning for MicroRNA target detection. **Journal of Biomedical Science and Engineering**, Citeseer, v. 3, n. 03, p. 247, 2010.

ZAHA, A.; FERREIRA, H. B.; PASSAGLIA, L. M. **Biologia Molecular Básica-5**. [S.l.]: Artmed Editora, 2014.