# Aplicação de técnicas de Aprendizado de Máquina para diagnóstico da doença Trombose Venosa Profunda

Rosana Guimarães Ribeiro
10 de Janeiro de 2018

## 1 DEFINIÇÃO

### 1.1 VISÃO GERAL

A trombose venosa profunda é uma doença que afeta milhares de pessoas em todo o mundo. Esta doença é caracterizada pelo surgimento de um coágulo de sangue – o trombo – em um vaso do sistema venoso profundo, o que determina a obstrução parcial ou total dessa veia e impede a circulação local. Esse evento pode afetar qualquer parte do corpo, mas acomete com mais frequência as veias das pernas e coxas, às vezes de forma associada a uma inflamação venosa – a flebite –, quando recebe a denominação de tromboflebite. A formação do coágulo deriva de diversos fatores de risco, especialmente da imobilidade prolongada, sendo mais comum em pessoas com idade superior a 50 anos.

A trombose pode ser superficial ou profunda, como a trombose venosa profunda. Entretanto, em qualquer dos casos o tratamento com medicação deve ser urgente, porque o coágulo de sangue pode fluir através da corrente sanguínea alojando-se em órgãos como os pulmões, gerando uma embolia pulmonar, ou no cérebro, gerando uma trombose cerebral, por exemplo, situações graves que podem até levar à morte [1].

A trombose venosa profunda (TVP) é motivo de atenção especial dado seu elevado risco relativo e absoluto e importante morbimortalidade. Neste contexto, os exames radiológicos assumem papel propedêutico na abordagem da enfermidade.

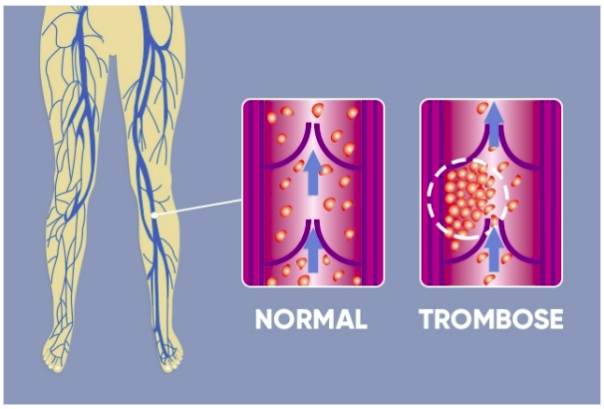



Figura 1.1: Trombose Venosa Profunda (TVP)

## 1.2 Problema

O diagnóstico clínico da TVP pode ser difícil, pois grande parte dos casos é assintomática, e também devido ao fato de que 30 a 50% dos pacientes com sintomas e sinais de TVP não apresentam a doença. A sintomatologia é comum a outras afecções e, assim, apenas os sinais clínicos não são suficientes para o diagnóstico de TVP.

Embora alguns pacientes procurem o médico com um quadro clínico bastante evidente, existem condições onde o diagnóstico pode ser mais difícil. Mesmo assim, o cirurgião vascular, utilizando dados da história pessoal e familiar, análise de fatores de risco conhecidos, exame físico, probabilidades de outros diagnósticos de outras doenças e a avaliação por meio de um índice de score, são capazes de classificar o paciente num grupo de baixa, média ou alta probabilidade de ter trombose venosa profunda.

A detecção e o reconhecimento de doenças com base em sistemas de aprendizagem de máquina podem fornecer indícios para identificar e tratar as doenças em seus estágios iniciais. Comparativamente, a identificação visual de doenças requer a experiência de um profissional da área, no modo geral, é cara, difícil e algumas vezes ineficiente.

A detecção automática de doenças, não somente da trombose é um tópico de pesquisa essencial, pois pode resultar em benefícios no monitoramento de grandes campos da área de saúde, e assim detectar automaticamente os sintomas de doenças logo que aparecem. Assim sendo, procurar um método rápido, automático, menos dispendioso e preciso para detectar casos da doença trombose é de grande significado realista.

O objetivo é criar uma aplicação baseada em técnicas de machine learning a fim de detectar e prever o grau de trombose de um paciente; as tarefas envolvidas são as seguintes:

1. Download, análise e pré-processamento dos dados TSUM_B.CSV
2. Treinar diferentes classificadores que possam determinar com maior precisão o grau de trombose
3. Análise dos resultados através das métricas adotadas e escolha do melhor modelo

Portando, visando a criação de uma aplicação para auxiliar os médicos no diagnóstico da doença trombose, este projeto toma como base para diagnóstico desta doença um conjunto de dados tendo como formato de entrada, dados **numéricos** e **categóricos** e saída esperada **quatro categorias**, sendo esta multi-classe. Sendo assim, a tarefa de aprendizagem aqui abordada faz uso de classificação multi-classe.

## 2 MÉTRICAS

Acurácia é a métrica de avaliação mais comum para os problemas de classificação, isto leva em consideração True Positives (TP) e True Negative (TN). Esta métrica também é a mais mal utilizada, ela é realmente adequada quando há um número igual de observações em cada classe (o que raramente é o caso) e que todas as previsões e erros de predição são igualmente importantes, o que geralmente não é o caso.

Para este projeto a acurácia será utilizada como métrica comum de análise, entretanto devido ao fato da base de dados estar desbalanceada e ser *multi-class*, outras métricas serão utilizadas para identificar a precisão do modelo.

$$acurácia = \frac{TP + TN}{TOTAL} \tag{2.1}$$

As métricas de suma importância para avaliação da aplicação são: *Recall*, precisão e f1. Precisão é a proporção de indivíduos que apresentam a doença corretamente previstas para o total de indivíduos que apresentam a doença prevista. Recall é a proporção de indivíduos que apresentam a doença corretamente previstas para todas as indivíduos que apresentam a doença e aqueles que não apresentam a doença corretamente previstas. *F1 Score* é a média ponderada de precisão e *recall*.

$$precisão = \frac{TP}{TP + FP} \tag{2.2}$$

$$Recall = \frac{TP}{TP + TN} \tag{2.3}$$

$$F1 = 2.\frac{precision.recall}{precision + recall} \tag{2.4}$$

Para construção desta aplicação, as métricas citadas acima ajudam a descobrir a qualidade do modelo através de acertos e erros com relação ao valor real e o valor previsto. Além disso, para visualizar os resultados graficamente, este projeto também adotará métricas como matriz de confusão.

# 3 ANÁLISE

## 3.1 EXPLORAÇÃO DE DADOS E VISUALIZAÇÃO

A base de dados utilizada neste projeto, TSUM_B.CSV, foi disponibilizada pelo hospital da Universidade de Chiba, Japão e pode ser encontrada no site de *Principles and Practice In Knowledge Discovery from Databases* através do link: http://lisp.vse.cz/pkdd99/Challenge/chall.htm. O conjunto de dados tem no total 805 instâncias, as quais são informações médicas e laboratoriais medidos pelo Laboratório de Doenças Colágenas de modo a classificar o paciente num grupo de baixa, média ou alta probabilidade de ter trombose venosa profunda.

A base de dados possui a seguinte característica:

- **ID**: identificação do paciente (int)
- **Examination Date**: data em que foi realizada o teste (int)
- **aCL IgG**: concentração de anticorpos IgG anticardiolipina (float)
- **aCL IgM**: concentração de anticorpos IgM anticardiolipina (float)
- **ANA**: concentração de anticorpos anti-núcleo (float)
- **ANA Pattern**: padrão observado na folha de exame ANA (string)
- **aCL IgA**: concentração de anticorpos IgA anticardiolipina (float)
- **Diagnosis**: nomes de doenças, atributo multivalorado (float)
- **KCT**: medida do grau de coagulação (string)
- **RVVT**: medida do grau de coagulação (string)
- **LAC**: medida do grau de coagulação (string)
- **Symptoms**: outros sintomas observados (string)
- **Thrombosis**: grau de thrombosis (int)
    - 0: negativo (sem trombose)
    - 1: positivo (o mais severo)
    - 2: positivo (severo)
    - 3: positivo (suave)

O Gráfico 3.1 mostra a quantidade total de instâncias para cada classe. Sendo 726 instâncias relacionadas à Classe 0, 56 à Classe 1, 18 à Classe 2 e apenas 5 à Classe 3. Como pode ser visto, o conjunto de dados esta desbalanceado e isto pode ocasionar medições errôneas ao aplicar um algoritmo de classificação tradicional, como SVM, Decision Tree ou Naive Bayes.

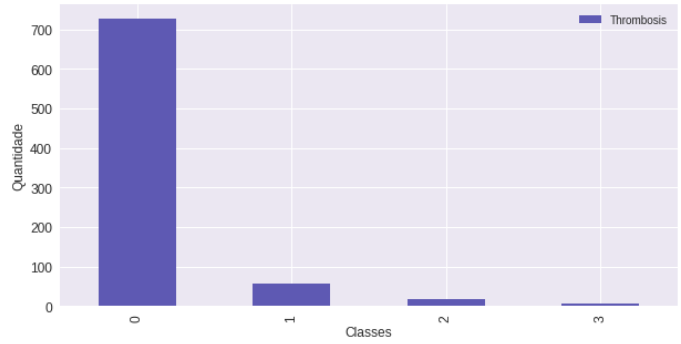



Figura 3.1: Conjunto de dados

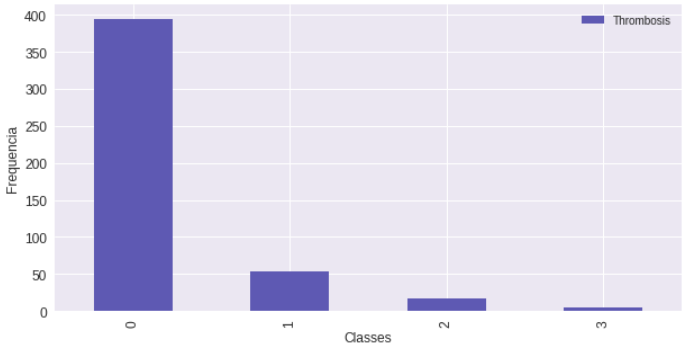



Figura 3.2: Conjunto de dados reduzidos

Como boa parte das instâncias fazem parte da Classe 0, foram deletadas àquelas em que o atributo Diagnosis apresenta valor nulo. Isso proporcionou na redução de boa parte das instâncias relacionadas à Classe 0, como pode ser visto no gráfico 3.2

Agora, o conjunto de dados apresenta 470 instâncias no total, com 394 pertencentes à Classe 0, 54 à Classe 1, 17 à Classe 2 e 5 à Classe 3. O processo de treinamento será feito nesta base de dados com o intuito de reduzir o desbalanceamento dos dados e auxiliar os algoritmos de classificação.

## 3.2 Algoritmos e Técnicas

A classificação é uma tarefa importante de reconhecimento de padrões. Uma variedade de algoritmos de aprendizagem de classificação, como a árvore de decisão, a rede neural de backpropagação, a rede bayesiana, o vizinho mais próximo, as máquinas de vetor de suporte e a classificação associativa recém-relatada, são bem desenvolvidas e aplicadas com sucesso em muitos domínios. No entanto, a distribuição de classes em um conjunto de dados desbalanceados encontra uma certa dificuldade para a maioria dos algoritmos de aprendizagem de classificação [2].

Dados desbalanceados são caracterizados por apresentar uma quantidade muito grande de uma determinada classe e uma quantidade de dados muito pequena pertencentes a outra

classe, quando tratando-se de classificação binária. Para este projeto, há dois problemas a serem abordados, os dados além de serem bastante desbalanceados, há o tratamento de multi-classes. Nessas condições, modelos de classificação que são otimizados em relação à precisão têm tendência de criar modelos triviais, que quase sempre predizem a classe majoritária.

Existem algotimos e métodos que trabalham de forma eficiente em conjunto de dados desbalanceados, alguns deles serão utilizados neste projeto.

### 3.2.1 AdaBoost

AdaBoost é um algoritmo de classificação que tem como idéia básica do aprendizado construir vários classificadores a partir dos dados originais e, em seguida, agregar suas previsões ao classificar amostras desconhecidas. A principal motivação para combinar classificadores em conjuntos redundantes é melhorar sua capacidade de generalização: cada componente classificador é conhecido por cometer o erro de ter sido treinado em um conjunto limitado de dados. Os padrões que são mal classificados pelos diferentes classificadores, no entanto, não são necessariamente os mesmos [2].

O AdaBoost é utilizado junto a algoritmos base para cruzar duas ou mais informações sobre dados analisados e, ao comparar com a base de dados de treinamento, reconhecer padrões e fazer as classificações. Sendo assim, suponha que duas informações diferentes (x e y) foram coletadas sobre diversos objetos de uma amostra. Ao serem analisadas por um algoritmo já treinado previamente, o resultado final adquirido será a classificação automática de acordo com os padrões encontrados e comparados com os dados de treinamento. Este resultado pode servir de entrada para uma tomada de decisão da própria máquina ou do controlador dela [3].

### 3.2.2 Resampling

Esta é uma técnica bastante comum em dados desbalanceados, como por exemplo em modelos para detecção de fraude, em um conjunto de transações, existem muitos usuários mais legítimos do que usuários fraudulentos. A técnica de Resampling consiste em igualar o conjunto de dados de cada classe, para o mesmo valor do conjunto de dados da classe minoritária. Assim, reduzindo as classes para a mesma quantidade, qualquer algoritmo de classificação pode ser aplicado para aprendizagem.

### 3.2.3 NearMiss

É um algoritmo baseado em Undersampling, uma das mais comuns e simples estratégias para dados desbalanceados tratando-se à classe majoritária. Remove instâncias da classe majoritária para diminuir o número de exemplos dela. É utilizado a biblioteca imbalanced learn que é compativel com o sklearn para fazer isso.

### 3.2.4 SMOTE

Esta é uma abordagem de sobre-amostragem na qual a classe minoritária é superestimada criando exemplos "sintéticos", em vez de uma sobre-amostragem com substituição. É gerado

exemplos sintéticos de uma maneira menos específica de aplicação, operando em "espaço de recursos"em vez de "espaço de dados". A classe minoritária é superestimada tomando cada amostra de classe minoritária e introduzindo exemplos sintéticos ao longo dos segmentos de linha juntando todos os vizinhos mais próximos da classe minoritária. Dependendo da quantidade de excesso de amostragem necessária, os vizinhos dos vizinhos mais próximos são escolhidos aleatoriamente.

#### 3.2.5 ADASYN

ADASYN é um módulo python que implementa uma técnica adaptativa de sobreamostragem para conjuntos de dados distorcidos. A idéia essencial do ADASYN é usar uma distribuição ponderada para diferentes exemplos de classes minoritárias de acordo com seu nível de dificuldade na aprendizagem, onde mais dados sintéticos são gerados para exemplos de classes minoritárias que são mais difíceis de aprender em comparação com os exemplos minoritários que são mais fáceis de aprender. Como resultado, a abordagem ADASYN melhora a aprendizagem em relação à distribuição de dados de duas formas: (1) redução do viés introduzido pelo desequilíbrio da classe e (2) mudança adaptativa do limite da decisão de classificação para os exemplos difíceis.

### 3.3 BENCHMARK

Este projeto tem como benchmark o modelo ao qual usa o algoritmo base do AdaBoost. Este modelo em que faz uso de diversos classificadores DecisionTree é o benchmark e os demais algoritmos são parte da solução, isto é, serão comparadas as pontuações dos algoritmos Resampling, NearMiss, SMOTE e ADASYN, que devem melhorar à medida que as técnicas mais avançadas são introduzidas e definir qual desses algoritmos possui o melhor resultado. Em seguida, é feita a comparação do algoritmo selecionado com o benchmark (algoritmo AdaBoost). Tal pontuação é determinada pelas métricas (f1, recall e precisão). Através dos resultados obtidos, será definido o modelo ótimo como a melhor referência. Portanto, o algoritmo base do AdaBoost possui acurácia de 83.2%, possui precisão de 91%, 57%, 29% e 60%, recall de 89%, 50%, 40% e 100%, por fim, f1 de 89%, 53%, 33% e 75% referentes às classes 0,1,2 e 3, respectivamente.

### 3.4 METODOLOGIA

#### 3.4.1 PRÉ-PROCESSAMENTO

O pré-processamento consistiu em modelar o conjunto de dados inicialmente desestruturado, em uma base de dados estruturada e adequada para aplicação de algoritmo de aprendizagem de classificação. Os seguintes critérios foram utilizados:

- Transformar atributos do tipo objeto em float
- Excluir atributos considerados desnecessários ao treinamento dos dados

- Fazer uso de Variáveis Dummy: transformar variáveis categóricas em variáveis numéricas
- Excluir instãncias em que há variáveis nulas no atributo Diagnosis
- Transformar valores simbólicos em numéricos
- Em demais variáveis nulas, atribuir a imputação média. Isto é, para cada atributo do tipo float em que há valor nulo, aplicar a média geral desses valores com relação ao atributo. Assim, valores nulos não serão descartados, mas sim transformados em uma média de todos os valores daquele atributo.
- Normalização dos dados. Para os atributos 'aCL IgG', 'aCL IgM', 'ANA', 'aCL IgA', é feita a normalização dos dados entre -1 e 1.
- Para aplicação de algoritmos baseados em Oversampling, é necessário dobrar o número da classe minoritária. Portanto, é feito uma cópia da classe 3 e dobrado o número de ínstâncias de 5 para 10. Assim, é possível aplicar algoritmos de aprendizagem em Oversampling.

### 3.4.2 Implementação

A implementação foi dividida em quatro partes principais:

1. **Exploração dos dados:** visualização e conhecimento dos dados
2. **Pré-processamento:** excluir instâncias e atributos desnecessários. Normalização de dados em que atributo possui valores muito altos. Uso de variável Dummy, transformar dados categóricos em numéricos.
3. **Treinamento dos dados:** Para os cinco algoritmos utilizados, 70% dos dados foram utilizados para treinamento, os demais 30% para teste. A quantidade de dados de treinamento permanece igual em todos os algoritmos, exceto quando usada a técnica de Resampling. Como nesta técnica a quantidade de instâncias de todas as quatro classes é reduzida para um valor semelhante à quantidade de instâncias da classe minoritária. Isto reduz a quantidade total de dados, entretanto essa redução é feita somente aos dados de treinamento, pois o conjunto de dados para teste permanece com a mesma quantidade que os demais algoritmos. Lembrando também que, os dados de treinamento utilizados na técnica Resampling são selecionados de forma randômica.

   Além disso, para os algoritmos exclusivos à dados desbalanceados, como NearMiss, SMOTE e ADASYN. Foi utilizado a técnica de pipeline. Técnica esta já apresentada como função pelo scikit-learn. O pipeline é o que encadena vários passos juntos, uma vez que a exploração inicial está concluída. Por exemplo, alguns códigos destinam-se a transformar características - normalizar numerais ou transformar texto em vetores, ou preencher dados faltantes, são transformadores. Outros códigos destinam-se a prever variáveis ajustando um algoritmo, como a Random Forest ou SVM, são estimadores.

Pipeline encadeia todos esses conjuntos, que podem então ser aplicados em dados de treinamento em bloco.

4. **Análise de resultados:** basea-se nas métricas f1, recall, precision e visualização através da matriz de confusão. Ao final, será escolhido o modelo ótimo como a melhor referência.

### 3.5 Refinamento

Como mencionado em [2], os algoritmos de Ensemble tem por motivação combinar classificadores em conjuntos redundantes, melhorando sua capacidade de generalização. Por isso, um dos algoritmos utilizados é o AdaBoost.

A fim de aprimorar o modelo utilizou-se a técnica de Resampling, que além de ser um refinamento por igualar os dados das múltiplas classes no processo de treinamento, este também faz uso do algoritmo AdaBoost. Isto é, inicialmente o AdaBoost é treinado no conjunto total de dados, em seguida, este mesmo algoritmo é utilizado em uma nova técnica em que os dados são reduzidos à classe minoritária, com o intuito de obter um modelo ótimo.

Houve refinamento em busca pelos melhores hiperparâmetros para o algoritmo de aprendizagem. Tais parâmetros são especificados abaixo conforme a sequência dos Algoritmos e Técnicas:

1. AdaBoost: n_estimators=150 e learning_rate=1. Para n_estimators foram utilizados valores como 10, 50 e 100, 150, 200, o que apresentou as melhores resultados para as métricas foi o de valor 150, após este valor os resultados das métricas decrescem.

2. Random Undersampling: por meio do parâmetro *ratio* é feita a variação das classes 0,1,2 e 3 para o cunjunto de treinamento. Através de várias combinações de random under sampling são analisadas as mudanças nos resultados da matriz de confusão quando aplicado o classificador da Árvore de Decisão. O código abaixo mostra o valor ideal dos parâmetros em *ratio*:

   ratio={ 0: y_train_dist_original[0]/22,

   1: y_train_dist_original[1]/3,

   2: y_train_dist_original[2],

   3: y_train_dist_original[3]}

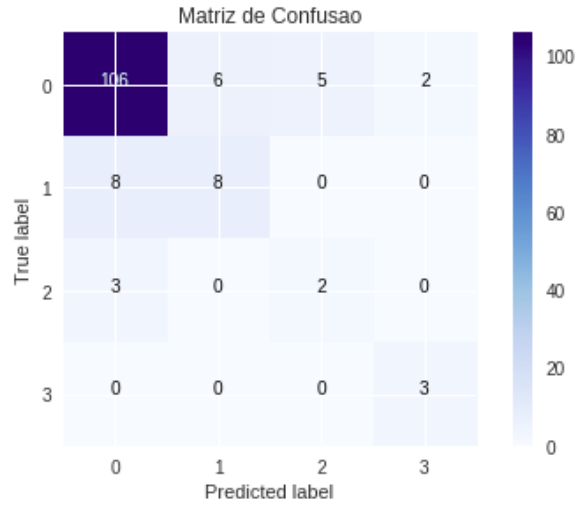
   Neste caso, a distribuição do treinamento após o resample a fim de alcançar hiperparâmetros ótimos são para cada classe:

   {0: 12, 1: 12, 2: 12, 3: 7}

## 4 Resultados

Segundo [2], algoritmos boosting são bons algoritmos quando trata-se de dados desbalanceados por construir vários classificadores através dos dados originais e depois agregam suas

previsões ao classificar amostras desconhecidas. A principal motivação para combinar classificadores em conjuntos redundantes é melhorar sua capacidade de generalização: cada componente classificador é conhecido por cometer o erro de ter sido treinado em um conjunto limitado de dados. Os padrões que são mal classificados pelos diferentes classificadores, no entanto, não são necessariamente os mesmos. O efeito da combinação de conjuntos redundantes também é estudado em termos de conceitos estatísticos de viés e variância. Dado um classificador, a decomposição de variância de polarização distingue entre o erro de polarização, o erro de variância e o erro intrínseco. Por tais motivos, foi aplicado o algoritmo AdaBoost para o conjunto de treinamento deste projeto, os resultados podem ser visto na Figura 4.1.



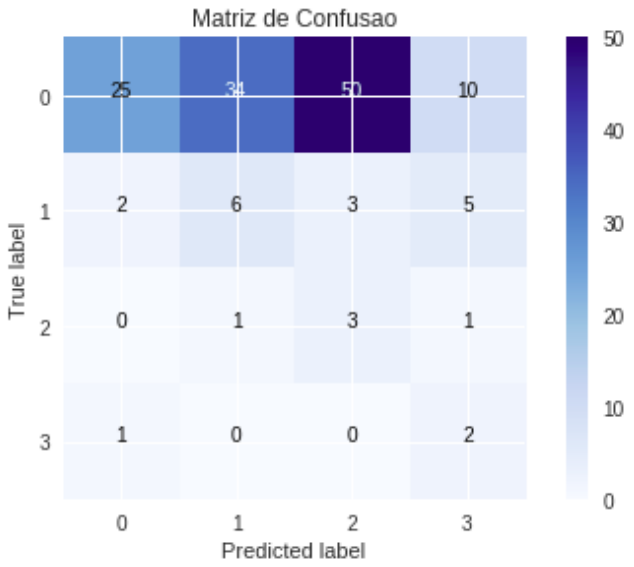

```
AdaBoost
accuracy: 0.832167832168
precision: [ 0.91  0.57  0.29  0.6 ]
recall: [ 0.89  0.5   0.4   1.  ]
f1: [ 0.9   0.53  0.33  0.75]
```

Figura 4.1: AdaBoost

O treinamento utilizando o algoritmo AdaBoost apresenta alta acurácia, mas enquadra-se como os algoritmos tradicionais. Ao criar um modelo com maior capacidade de generalização, o algoritmo de aprendizagem foca na classe majoritária (classe 0), e não tanto nas demais classes. Ainda assim, para a classe minoritária, há um valor de f1 com 75%, um bom valor apesar da Classe 3 possuir apenas 5 instâncias.

O NearMiss, por ser um algoritmo baseado em Undersampling, remove instâncias da classe majoritária para diminuir o número de exemplos, e tentar igualar o aprendizado para as quatro classes. Além de apresentar um valor de acurária bem baixo, as métricas como recall, f1 e precisão, também não tiveram valores altos, apenas para a precisão da classe 0. Figura 4.2.

O método Resample é bastante eficaz para dados desbalanceados. O que este método faz é reduzir o número de todas as classes para o valor do número da classe minoritária. Assim,



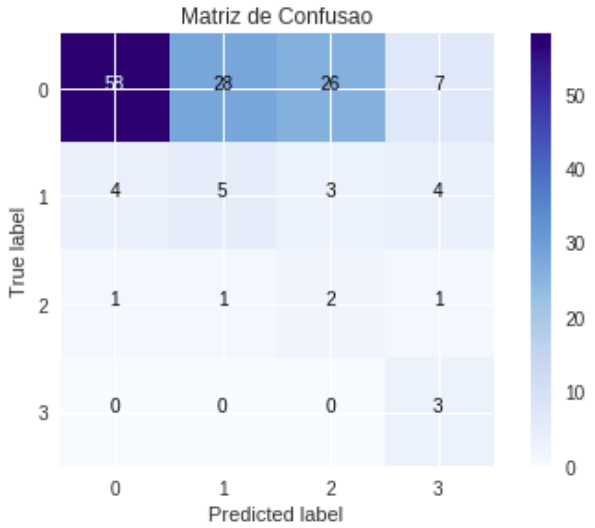

```
NearMiss
accuracy: 0.251748251748
precision: [ 0.89  0.15  0.05  0.11]
recall: [ 0.21  0.38  0.6   0.67]
f1: [ 0.34  0.21  0.1   0.19]
```

Figura 4.2: NearMiss

depois dessa redução, qualquer algoritmo de aprendizagem pode ser utilizado. Neste caso, foi utilizada a Árvore de Decisão. Os resultados através das métricas adotadas pode ser vista na 4.3.

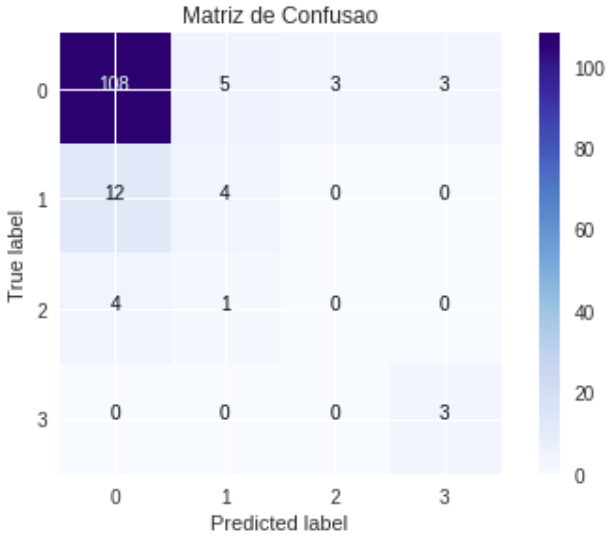
Para o uso dos algotimos de Oversampling, SMOTE e ADASYN, é necessário que o total de instâncias da classe minoritária esteja balanceada em relação ao total de instâncias das demais classes, caso contrário, é impossível fazer uso desses algotimos. Devido a isto, na etapa de pré-processamento, foi feito um aumento do número de instâncias da Classe 3, de 5 para 10. As demais classes permaneceram com o mesmo número de instâncias, não houve modificações.

SMOTE é um algoritmo apropriado para dados desbalanceados. Este cria dados sintéticos da classe minoritária, esta que no momento de aprendizagem é super-estimada. O SMOTE apresenta boa precisão para as Classes 0, 1 e 3. Entretanto, não houve alguma precisão, recall ou f1 para a classe 2, ver Figura 4.4.




```
Decision Tree + Random Undersampling
accuracy: 0.475524475524
precision: [ 0.92  0.15  0.06  0.2 ]
recall: [ 0.49  0.31  0.4   1.  ]
f1: [ 0.64  0.2   0.11  0.33]
```

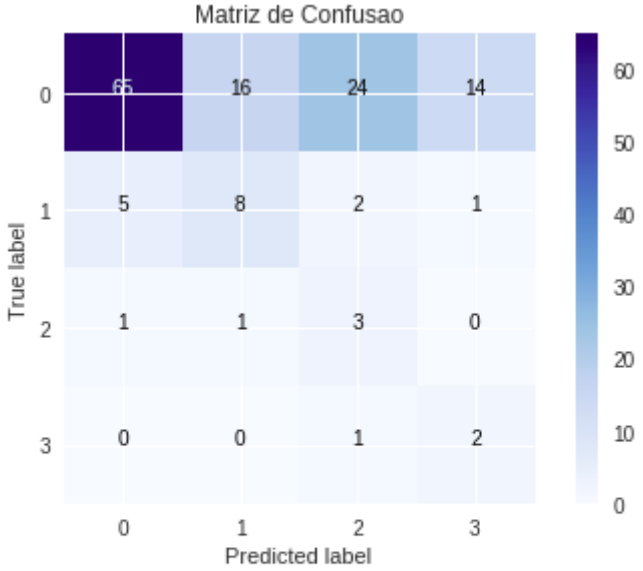
Figura 4.3: Decision Tree + Random Undersampling




```
SMOTE
accuracy: 0.804195804196
precision: [ 0.87  0.4   0.    0.5 ]
recall: [ 0.91  0.25  0.    1.  ]
f1: [ 0.89  0.31  0.    0.67]
```

Figura 4.4: SMOTE

ADASYN é um método que trabalha de forma semelhante ao SMOTE. O resultado pode ser visto com acurácia acima de 54%, f1 com score de 68% para a classe 0, enquanto para as demais classes esse valor fica abaixo de 50%, ver Figura 4.5.




```
ADASYN
accuracy: 0.545454545455
precision: [ 0.92  0.32  0.1   0.12]
recall: [ 0.55  0.5   0.6   0.67]
f1: [ 0.68  0.39  0.17  0.2 ]
```

Figura 4.5: ADASYN

# 5 Conclusão

O presente projeto tem como objetivo o reconhecimento da doença Trombose Venosa Profunda com base em sistemas de aprendizagem de máquina. A aplicação detecta diferentes graus da doença, dividido entre negativo (sem trombose) e positivo (mais severo, severo, suave).

A classificação dos dados desbalanceados continua a receber cada vez mais atenção no mundo científico e industrial. A aplicação apresenta algumas dificuldades em encontrar o melhor score devido ao desbalanceamento dos dados com multi classes. A combinação entre múltiplas classes e dados desbalanceados assume uma situação bastante complexa. Tal situação desafia o paradigma clássico de reconhecimento de padrões e exige um tratamento diferente.

Dentre todos os modelos de classificação exibidos neste projeto, a classificação que consegue generalizar o problema é o AdaBoost. A técnica de Resample também é bastante eficaz, reduzir o conjunto de dados para uma quantidade próxima da classe minoritária contribui

no processo de aprendizagem, pois trata o conjunto de forma igualitária. Entretanto, utilizar um algoritmo que tem como idéia básica do aprendizado construir vários classificadores e agregar suas previsões a fim de minimizar o erro, ainda assim é a melhor solução. Os demais métodos apesar de serem eficazes em dados desbalanceados, não apresentam bons resultados para o reconhecimento da doença TVP. Para este problema e pela base de dados adotada, é necessário a coleta de mais dados para as Classes 1,2 e 3, assim, os modelos de generalização podem apresentar melhor eficiência nas métricas.

## 6 Trabalhos Futuros

A maioria das pesquisas em que explora problemas de dados desbalanceados, tratam-se de classes binárias. Há poucos relatos sobre dados desbalanceados para multi-classe. Isso se deve ao fato de que os dados desbalanceados de classe binária são penetrantes em um grande número de projetos práticos de grande importância, como por exemplo, detecção de câncer, definir se um paciente tem ou não a doença, detecção de transações fraudulentas, definir se há ou não fraude, dentre outros projetos. Então, se trabalhar com multi-classes já é um papel difícil, a situação se torna mais complicada quando estes dados estão desbalanceados. Portanto, segue como uma pesquisa interessante para investigação futura, a classificação de dados desbalanceados com rótulos de classe múltipla. Tentar descobrir qual o melhor algoritmo e técnica a serem utilizados e de que forma este algoritmo possa ser refinado para alcançar um modelo ótimo.

## 7 Referências Bibliográficas

[1] Tua Saúde. **Como identificar a trombose e o que fazer para evitar.** Website: . Acesso em 29 de Novembro de 2017.

[2] WONG, A. K. C. **Classification of imbalanced data: a review**. World Scientific Publishing Company. International Journal of Pattern Recognition and Artificial Intelligence. Vol. 23, No. 4. p.p 687–719. 2009.

[3] CHAVES, B.B. **Estudo do Algoritmo AdaBoost de Aprendizagem de Máquina Aplicado a Sensores e Sistema Embarcados**. Dissertação de Mestrado pela Politécnica da Universidade de São Paulo. São Paulo, 2012.