Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Avaliação de Algoritmos de Aprendizagem de Máquinas na Detecção de Mutações Somáticas

Pedro Aurélio Coelho de Almeida

Monografia apresentada como requisito parcial
para conclusão do Curso de Engenharia da Computação

Orientador
Prof. Dr. Díbio Leandro Borges

Brasília
2019

Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Avaliação de Algoritmos de Aprendizagem de Máquinas na Detecção de Mutações Somáticas

Pedro Aurélio Coelho de Almeida

Monografia apresentada como requisito parcial
para conclusão do Curso de Engenharia da Computação

Prof. Dr. Díbio Leandro Borges (Orientador)
CiC/UnB

Prof. Dr. Jan Mendonça Correa
CiC/UnB

Profª. Drª. Roberta Barbosa Oliveira
CiC/UnB

Prof. Dr. José Edil Guimarães de Medeiros
Coordenador do Curso de Engenharia da Computação

Brasília, 14 de Novembro de 2019

Dedicatória

Agradeço, inicialmente, à natureza e à realidade, pois, se elas fossem diferentes, as coisas poderiam não ser tão divertidas. Dedico esse trabalho à minha família, especialmente aos meus pais que me ajudaram muito em todo o processo.

À Natália, minha namorada, quero agradecer todo o carinho, paciência e apoio que tive durante o curso. Não posso deixar de agradecer imensamente aos meus amigos, especialmente os mais próximos (Rodrigo, Valerie e Matheus), por terem me dado apoio.

Por fim, tenho enorme gratidão aos meus professores, especialmente os Profs. Díbio, Zaghetto e a Profa. Flávia por terem me ensinado o caminho da ciência e a necessidade de nunca desistir, mesmo com as dificuldades da vida.

Agradecimentos

Agradeço ao Laboratório de Imagens, Sinais e Áudio (LISA) por ceder a máquina utilizada para executar parte dos experimentos e ao meu orientador Prof. Dr. Díbio Leandro Borges pela paciência, sugestões e incansável trabalho de orientação.

O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES), por meio do Acesso ao Portal de Periódicos.

Resumo

O estudo do DNA é importante para aplicações clínicas e de pesquisa. Dada a complexidade de sua análise, o uso de ferramentas computacionais se torna extremamente vantajoso. Este trabalho compara o desempenho dos modelos de aprendizagem de máquinas (*Isolation* e *Random Forest*) na detecção de mutações somáticas (inserções/remoções e polimorfismo de nucleotídeo único), utilizando os atributos calculados pela ferramenta chamada *Strelka2*. O treinamento dos modelos foi realizado por meio de validação cruzada do tipo *k-fold* (k=10) com avaliação das métricas de revocação e *f1-score* nas bases de testes utilizadas pelos autores do *Strelka2*. A partir destes testes, o *Random Forest* apresentou *f1-scores* maiores que 0,9, enquanto que o *Isolation Forest* resultou em valores inferiores a 0,75. Utilizar o *Random Forest* é recomendável quando se tem dados rotulados e se deseja alta revocação e precisão. Investigações futuras incluem a avaliação de outras técnicas de aprendizagem não supervisionada como mapas auto organizáveis e/ou uso de diferentes atributos para o *Isolation Forest*.

Palavras-chave: SNV, *Indels*, Detecção de mutações somáticas, *Strelka*2, *Isolation Forest*, *Random Forest*

Abstract

DNA analysis is very important for clinical and research purposes. Given its complexity, computers become useful tools. This work presents a comparison between both the Isolation and Random Forest machine learning techniques using the features calculated by the somatic mutation caller Strelka2 for both insertions/deletions and single nucleotide variants. Both models were calibrated using k-fold cross-validation (k=10) and evaluated considering recall and f1-score metrics for the test bases used by Strelka2. From these trials, Random Forest reached f1-scores greater than 0.9 while Isolation Forest presented values lower than 0.75 for the same metric. Using Random Forest is recommended when there is labeled data and when one requires high precision and recall. Future research would include evaluating different unsupervised learning models namely self organizing maps and/or using a different feature set to calibrate Isolation Forest.

Keywords: SNV, Indels, somatic mutation calling, Strelka2, Isolation Forest, Random Forest

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas e Siglas

ADN Ácido DesoxirriboNucleico.

ADs Árvores de Decisão.

AM Aprendizagem de Máquinas.

ANN *Aritificial Neural Newtroks*.

BAM *Binary Alignment Map*.

BED *Browser Extensible Data*.

CNN *Convolutional Neural Newtroks*.

DNA *DeoxyriboNucleic Acid*.

FN Falso Negativo.

FP Falso Positivo.

IA Inteligência Artificial.

indels *Insertions/Deletions*.

SAM *Sequence Alignment Map*.

SNP *Single Nucleotide Polymorphism*.

SNV *Single Nucleotide Variant*.

VCF *Variant Call Format*.

VN Verdadeiro Negativo.

VP Verdadeiro Positivo.

Capítulo 1

Introdução

O estudo do *DeoxyriboNucleic Acid* (DNA, em inglês) ou Ácido DesoxirriboNucleico (ADN, em português) humano vem se mostrando fundamental e extremamente útil para a compreensão do corpo humano e de algumas patologias que possam acometê-lo. Descobertas como o mecanismo pelo qual as doenças se originam (como, por exemplo, de Huntington, que consiste na repetição exagerada da sequência de nucleotídeos do DNA - citosina, adenina e guanina ou *CAG* de forma abreviada - (VONSATTEL; DIFIGLIA, 1998)) deixam a humanidade mais próxima de soluções de tratamentos e, possivelmente, de cura para diversos males que assolam as pessoas.

Devido à enorme quantidade de nucleotídeos presentes no DNA (bilhões de pares) (AMABIS; MARTHO, 2006) e da variedade de combinações entre diferentes populações, o uso de ferramentas computacionais para distinguir bases que contêm mutações verdadeiras das que não as apresentam se torna extremamente vantajoso e atrativo. Dentre esses mecanismos, pode-se mencionar o *Neusomatic* (SAHRAEIAN et al., 2019), o *Muse* (FAN et al., 2016) e o *Strelka2* (KIM et al., 2018). Eles utilizam, respectivamente, abordagens puramente de Inteligência Artificial (IA), modelos puramente estocásticos e uma combinação entre IA e modelos estocásticos para realizar a tarefa de detectar mutações.

1.1 Objetivo geral

Nesse estudo, procurar-se-á utilizar dos recursos proporcionados pela área da IA na detecção de alterações (mutações) de DNA, visto que as abordagens que utilizam IA para resolver essa tarefa apresentam resultados melhores ou iguais aos obtidos por aquelas que somente utilizam modelos estocásticos como mostrado nos gráficos de comparação entre os modelos presente no artigo do *Neusomatic* (SAHRAEIAN et al., 2019).

1.2 Objetivos específicos

Para isso, comparar-se-ão algoritmos de Aprendizagem de Máquinas (AM) *Isolation Forest* (LIU; TING; ZHOU, 2008) e *Random Forest* (JAMES et al., 2013), em inglês ou Floresta Randômica em tradução livre, com o intuito de se verificar qual a melhor performance na detecção de mutações somáticas em células humanas. O classificador *Strelka2* (KIM et al., 2018) será utilizado como ponto de partida para a comparação, vez que fornecerá o conjunto de atributos (*features*, em inglês) de entrada para ambos classificadores e servirá, também, como um pré-filtro de possíveis mutações.

Além disso, uma otimização dos parâmetros do *Isolation Forest* será realizada a fim de obter as melhores configurações para a comparação.

1.3 Organização do trabalho

No Capítulo 2 será abordada a teoria que embasa o estudo. O Capítulo 3 apresenta os métodos e configurações dos computadores utilizados. O Capítulo 4 traz os resultados e uma breve discussão sobre eles. Finalmente, o Capítulo 5 conclui a pesquisa, resumindo as principais descobertas.

Capítulo 2

Fundamentação Teórica

Abordar-se-ão conceitos fundamentais nas áreas de genética e AM para que o leitor possa compreender as técnicas e termos utilizados nos demais capítulos.

2.1 DNA

DNA é o composto orgânico localizado no núcleo das células eucarióticas (AMABIS; MARTHO, 2006). É formado por uma longa sequência de nucleotídeos que se apresentam em pares (Adenina-Timina e Citosina-Guanina) e, a partir da decodificação de partes dessa sequência, as células do organismo produzem proteínas e regulam seu funcionamento (GRIFFITHS et al., 2000).

O DNA humano está contido em estruturas chamadas de cromossomos e é o responsável por modelar características anatômicas e fisiológicas no organismo humano. O gene é a porção de DNA que forma a unidade básica responsável por essa funcionalidade. Diferentes formas de genes localizados no mesmo *locus* são denominadas alelos, sendo os seres homozigóticos aqueles que possuem dois alelos iguais de um gene e heterozigóticos os que possuem alelos diferentes. Por fim, a frequência de alelos é definida como a proporção de todos os alelos de um tipo específico na população ((AMABIS; MARTHO, 2006) e (GRIFFITHS et al., 2000)).

Atualmente, máquinas de sequenciamento extraem essas sequências como cadeias de caracteres, onde cada caractere (A, C, G e T) representa um dos nucleotídeos. Devido ao tamanho do DNA (bilhões de nucleotídeos), taxas de erro muito pequenas como, por exemplo, 0.0001% podem acarretar em milhares de nucleotídeos sequenciados incorretamente. Por isso, uma boa prática é realizar diversos sequenciamentos na mesma leitura do DNA, aumentando a cobertura (MOORTHIE; MATTOCKS; WRIGHT, 2011) e criando, por exemplo, sequenciamentos com coberturas 100x, ou seja, o processo de sequenciar o

DNA para este exemplo foi realizado 100 vezes na mesma amostra para reduzir as chances de erros ocorrerem.

2.2 Mutações somáticas

Alterações na sequência original do DNA de um indivíduo são chamadas de mutações (AMABIS; MARTHO, 2006). Podem ocorrer por diversos motivos e afetar células germinativas (responsáveis pela reprodução dos indivíduos), as quais são haploides por possuírem somente um cromossomo em vez do par (AMABIS; MARTHO, 2006), e somáticas (células não germinativas) (GRIFFITHS et al., 2000), que são diploides por possuírem um par de cromossomos (AMABIS; MARTHO, 2006).

Mutações em células somáticas são particularmente interessantes em estudos relacionados ao tratamento de indivíduos, uma vez que em células germinativas afetarão os descendentes de uma pessoa e não a própria pessoa (GRIFFITHS et al., 2000). Devido a esse fato, as ferramentas computacionais utilizadas focarão somente na detecção de mutações somáticas. Foram estudados dois tipos de mutação: inserções/remoções de cadeias de nucleotídeos (*indels*, em inglês) e polimorfismo de nucleotídeo único (SNP ou SNV, em inglês). Esses últimos são processos nos quais somente um nucleotídeo é alterado (XU, 2018).

A detecção de mutações somáticas ocorre por meio do alinhamento de 3 sequenciamentos diferentes: um advindo de uma célula doente, outro de uma célula normal e uma terceira sequência de referência, criada por especialistas na tentativa de elaborar um modelo padrão do DNA humano, utilizada para alinhar as duas sequências anteriores (XU, 2018). A partir desse alinhamento, abordagens computacionais, como o uso de AM combinado com modelos estatísticos, podem ser empregadas para verificar quais posições e alterações são realmente mutações e não erros de sequenciamento.

2.3 Formatos de arquivos utilizados

O sequenciamento e alinhamento do DNA são gravados em formatos predefinidos. Existem formatos para armazenar sequências 'cruas' de nucleotídeos (FASTA), regiões de interesse de análise (BED), alinhamentos de várias sequências (SAM ou BAM) e formatos que descrevem a alteração (mutação) em termos da posição no cromossomo e sequências de referência e mutantes (VCF) (ZHANG, 2016).

2.3.1 FASTA

O FASTA é um formato de arquivo texto, cuja extensão é *.fa* ou *.fasta*. Ele contém uma sequência de caracteres correspondentes às bases de DNA a serem analisadas (A, T, C e G por exemplo). A Figura 2.1, extraída do manual de especificação do formato SAM[1], contém exemplos de 5 sequenciamentos, sendo um deles a referência, semelhantes ao encontrado em arquivos FASTA.

```
Coor     12345678901234  5678901234567890123456789012345
ref      AGCATGTTAGATAA**GATAGCTGTGCTAGTAGGCAGTCAGCGCCAT

+r001/1        TTAGATAAAGGATA*CTG
+r002         aaaAGATAA*GGATA
+r003       gcctaAGCTAA
+r004                     ATAGCT..............TCAGC
-r003                           ttagctTAGGC
-r001/2                                      CAGCGGCAT
```

Figura 2.1: Exemplo de arquivo em formato FASTA, contendo cinco leituras: r001/1, r002, r003, r004 e a sequência de referência.

2.3.2 BED

Arquivos *Browser Extensible Data* (BED, em inglês, cuja extensão é *.bed*) são delimitados por tabulação e possuem 3 campos obrigatórios que costumam ser os únicos utilizados: nome do cromossomo, a posição inicial de análise (a primeira base tem o índice 0) e posição final de análise. A Figura 2.2, extraída da página da Universidade da Califórnia Santa Cruz (UCSC, em inglês)[2], mostra um exemplo de arquivo BED que contém, dentre outros, os três primeiros campos obrigatórios (nome do cromossomo e posições iniciais e finais), a partir da quarta linha, e informações adicionais para exibição em formato RGB contendo 6 dos 9 campos adicionais (*name*, *score*, *strand*, *thickStart*, *thickEnd* e *itemRgb*).

[1]Link para acesso à página do manual de especificação do formato SAM do qual foi extraída a imagem de exemplo de sequências para ilustrar o formato FASTA *<https://samtools.github.io/hts-specs/SAMv1.pdf>*. Acessada em 17/10/2019

[2]Link para acesso à página da Universidade da Califórnia Santa Cruz (UCSC, em inglês), da qual se extraiu o exemplo de um arquivo no formato BED. *<https://genome.ucsc.edu/FAQ/FAQformat.html#format1>*. Acessada em 17/10/2019

```
browser position chr7:127471196-127495720
browser hide all
track name="ItemRGBDemo" description="Item RGB demonstration" visibility=2 itemRgb="On"
chr7    127471196  127472363  Pos1  0  +  127471196  127472363  255,0,0
chr7    127472363  127473530  Pos2  0  +  127472363  127473530  255,0,0
chr7    127473530  127474697  Pos3  0  +  127473530  127474697  255,0,0
chr7    127474697  127475864  Pos4  0  +  127474697  127475864  255,0,0
chr7    127475864  127477031  Neg1  0  -  127475864  127477031  0,0,255
chr7    127477031  127478198  Neg2  0  -  127477031  127478198  0,0,255
chr7    127478198  127479365  Neg3  0  -  127478198  127479365  0,0,255
chr7    127479365  127480532  Pos5  0  +  127479365  127480532  255,0,0
chr7    127480532  127481699  Neg4  0  -  127480532  127481699  0,0,255
```

Figura 2.2: Exemplo de arquivo em formato BED, contendo os três primeiros campos obrigatórios (nome do cromossomo e posições iniciais e finais), a partir da quarta linha, e informações adicionais para exibição em formato RGB.

2.3.3 SAM/BAM

Arquivos *Sequence Alignment Map* (SAM, em inglês, cuja extensão é *.sam*) possuem formato delimitado por tabulação e têm o objetivo de mapear alterações de uma dada leitura com relação a um genoma de referência. Arquivos *Binary Alignment Map* (BAM, em inglês, cuja extensão é *.bam*) são a versão compactada (binária) de arquivos SAM.

A Figura 2.3, extraída do manual de especificação do formato SAM[3], mostra um exemplo do alinhamento resultante dos sequenciamentos presentes na Figura 2.1. Para ilustrar melhor o processo de alinhamento de duas ou mais sequências de DNA, extraiu-se de Sahraeian et al. (2019) a Figura 2.4, a qual contém um exemplos de sequenciamentos fictícios de 3 amostras (referência, célula com tumor e célula normal) na imagem da esquerda e seus respectivos alinhamentos na imagem da direita.

```
@HD VN:1.6 SO:coordinate
@SQ SN:ref LN:45
r001   99 ref  7 30 8M2I4M1D3M = 37  39 TTAGATAAAGGATACTG *
r002    0 ref  9 30 3S6M1P1I4M *  0   0 AAAAGATAAGGATA    *
r003    0 ref  9 30 5S6M       *  0   0 GCCTAAGCTAA       * SA:Z:ref,29,-,6H5M,17,0;
r004    0 ref 16 30 6M14N5M    *  0   0 ATAGCTTCAGC       *
r003 2064 ref 29 17 6H5M       *  0   0 TAGGC             * SA:Z:ref,9,+,5S6M,30,1;
r001  147 ref 37 30 9M         =  7 -39 CAGCGGCAT         * NM:i:1
```

Figura 2.3: Exemplo de arquivo em formato SAM, gerado a partir das leituras presentes na Figura 2.1.

[3]Link para acesso à página do manual de especificação do formato SAM do qual foi extraída a imagem de exemplo de alinhamento para ilustrá-lo *<https://samtools.github.io/hts-specs/SAMv1.pdf>*. Acessada em 17/10/2019

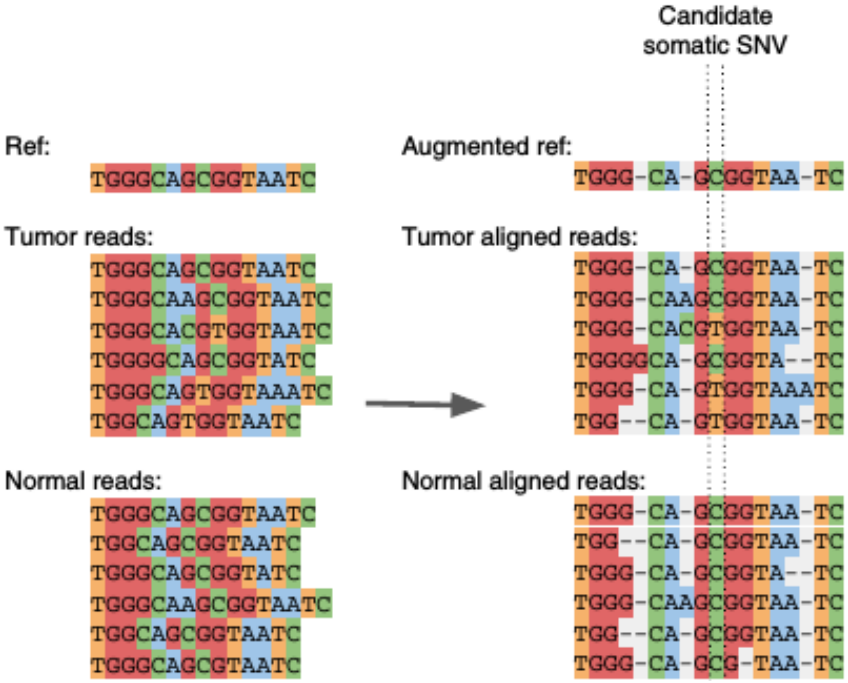


Figura 2.4: Ilustração do processo de alinhamento (imagem à direita) a partir de sequenciamentos (imagem à esquerda) de 3 amostras (referência, amostra com tumor e amostra normal). (Fonte: (SAHRAEIAN et al., 2019)).

2.3.4 VCF

Arquivos *Variant Call Format* (VCF, em inglês, cuja extensão é *.vcf*) também são separados por tabulação e têm o propósito de indicar variações em algum conjunto de dados. São muito usados por programas que detectam variações genômicas (*variant callers*, em inglês).

A Figura 2.5, extraída do manual de especificação do formato VCF[4], contém um exemplo de dados armazenados em um arquivo VCF. As linhas com *##* pertencem ao cabeçalho. A primeira linha do arquivo (*##fileformat*) indica a versão do VCF utilizada. As linhas *##INFO*, *##FILTER* e *##FORMAT* presentes no cabeçalho descrevem o significado das siglas e informações presentes, respectivamente, nas colunas *INFO*, *FILTER* e *FORMAT* da seção de dados. A seção de dados possui, em geral, além das colunas mencionadas, as seguintes:

- a indicação do número do cromossomo analisado em cada linha (*#CHROM*);
- a posição inicial na sequência de referência (a primeira base tem posição 1);
- o ID da sequência analisada ou '.' quando este não é especificado;

[4]Link para acesso à página do manual de especificação do formato VCF do qual foi extraída a imagem para ilustrá-lo *<http://samtools.github.io/hts-specs/VCFv4.2.pdf>*. Acessada em 17/10/2019.

- a sequência de bases presentes na referência;
- as sequências de bases alteradas que foram detectadas;
- um fator de qualidade na escala Phred ($-10log(erro)$).

Outras colunas são opcionais e podem não estar presentes em todos os arquivos VCF.

```
##fileformat=VCFv4.2
##fileDate=20090805
##source=myImputationProgramV3.1
##reference=file:///seq/references/1000GenomesPilot-NCBI36.fasta
##contig=<ID=20,length=62435964,assembly=B36,md5=f126cdf8a6e0c7f379d618ff66beb2da,species="Homo sapiens",taxonomy=x>
##phasing=partial
##INFO=<ID=NS,Number=1,Type=Integer,Description="Number of Samples With Data">
##INFO=<ID=DP,Number=1,Type=Integer,Description="Total Depth">
##INFO=<ID=AF,Number=A,Type=Float,Description="Allele Frequency">
##INFO=<ID=AA,Number=1,Type=String,Description="Ancestral Allele">
##INFO=<ID=DB,Number=0,Type=Flag,Description="dbSNP membership, build 129">
##INFO=<ID=H2,Number=0,Type=Flag,Description="HapMap2 membership">
##FILTER=<ID=q10,Description="Quality below 10">
##FILTER=<ID=s50,Description="Less than 50% of samples have data">
##FORMAT=<ID=GT,Number=1,Type=String,Description="Genotype">
##FORMAT=<ID=GQ,Number=1,Type=Integer,Description="Genotype Quality">
##FORMAT=<ID=DP,Number=1,Type=Integer,Description="Read Depth">
##FORMAT=<ID=HQ,Number=2,Type=Integer,Description="Haplotype Quality">
#CHROM POS     ID        REF    ALT     QUAL FILTER INFO                              FORMAT      NA00001        NA00002        NA00003
20     14370   rs6054257 G      A       29   PASS   NS=3;DP=14;AF=0.5;DB;H2           GT:GQ:DP:HQ 0|0:48:1:51,51 1|0:48:8:51,51 1/1:43:5:.,.
20     17330   .         T      A       3    q10    NS=3;DP=11;AF=0.017               GT:GQ:DP:HQ 0|0:49:3:58,50 0|1:3:5:65,3   0/0:41:3
20     1110696 rs6040355 A      G,T     67   PASS   NS=2;DP=10;AF=0.333,0.667;AA=T;DB GT:GQ:DP:HQ 1|2:21:6:23,27 2|1:2:0:18,2   2/2:35:4
20     1230237 .         T      .       47   PASS   NS=3;DP=13;AA=T                   GT:GQ:DP:HQ 0|0:54:7:56,60 0|0:48:4:51,51 0/0:61:2
20     1234567 microsat1 GTC    G,GTCT  50   PASS   NS=3;DP=9;AA=G                    GT:GQ:DP    0/1:35:4       0/2:17:2       1/1:40:3
```

Figura 2.5: Exemplo de arquivo em formato VCF.

2.4 Aprendizagem de máquinas

O Aprendizagem de Máquinas pode ser entendido como um ramo da IA. Procura, a partir da detecção de padrões em conjuntos de dados, criar modelos que automaticamente se adaptem à alguma tarefa. O objetivo desse processo é responder adequadamente ao receber um conjunto de dados anteriormente desconhecido ((MITCHELL, 1997) e (SHALEV-SHWARTZ; BEN-DAVID, 2014)).

2.4.1 Aplicações

Nos últimos anos, o uso de técnicas da área da IA, especialmente aquelas relacionadas ao AM, tem se mostrado eficiente na solução de diversos problemas, como no processamento de linguagem natural (ZHANG; ZHAO; LECUN, 2015) e na bioinformática (KIM et al., 2018).

Aplicações de AM possibilitam que computadores consigam traçar fronteiras que separam diferentes dados (classificados ou não). Quanto maior a quantidade de dados fornecida, melhores tendem a ser os resultados (JUN et al., 2007).

Considerando a necessidade e a vantagem de se detectar mutações somáticas, vários modelos de AM foram utilizados, como, por exemplo, *Convolutional Neural Newtroks*

(CNN), em inglês ou redes neuronais[5] convolucionais em tradução livre, (SAHRAEIAN et al., 2019); uma combinação de diferentes detectores de mutações com diversos métodos de AM (ANZAR et al., 2019) e Árvores de Decisão (ADs) (KIM et al., 2018). Os autores do *Neusomatic* utilizaram uma modificação da arquitetura *ResNet* (HE et al., 2016), enquanto que os autores do *Strelka2* empregaram o modelo de *Random Forest* (JAMES et al., 2013). Dentre esses, ADs são particularmente interessantes, uma vez que, se o conjunto de atributos conseguir separar bem os dados entre diferentes classes e/ou agrupamentos, então apresentarão bom desempenho nas métricas de avaliação (revocação, precisão e acurácia) e uma baixa complexidade para 'aprender' (treinar) e classificar ('predizer') novos dados.

2.4.2 Tipos de aprendizagem

Existem três grandes estilos de treinamento para AM ((LIBBRECHT; NOBLE, 2015) e (JAMES et al., 2013)):

- supervisionado: existe um conjunto de rótulos ou classes para os quais se deseja classificar os dados;
- não supervisionado: não há rótulos para classificação e, por isso, o objetivo é agrupar os dados de uma maneira que possa fazer sentido ou ser útil para alguma aplicação;
- semi-supervisionado: abordagem híbrida em que alguns dados de treinamento estão rotulados e outros não. Logo, utiliza-se inicialmente a porção com classes definidas para treinamento e, em seguida, classificam-se de forma não supervisionada os dados não rotulados e usam-se as classes atribuídas para retreinar o modelo.

Dentre as categorias acima, a aprendizagem supervisionada é utilizada em diversas tarefas, incluindo a detecção de mutações somáticas como feito, por exemplo, em Sahraeian et al. (2019) e Kim et al. (2018). Nesse caso, em geral, uma equipe de especialistas analisa alinhamentos de DNA de amostras conhecidas e estabelece quais regiões são efetivamente mutações.

Por causa da presença de rótulos-alvo, pode-se medir o desempenho de um classificador supervisionado em termos de métricas de acerto ou erro, como, por exemplo, acurácia, precisão e revocação (*recall*, em inglês). Para obter essas métricas, é comum dividir o conjunto de dados classificados entre base de treino e base de teste ou entre bases de treino, validação e teste. A primeira abordagem avalia o desempenho de um classificador treinado com a base de treino ao classificar os dados na base de teste, não usados durante

[5]Neste trabalho, o termo neuronais será usado como sinônimo tanto para redes convolucionais quanto para redes genéricas do tipo *Aritificial Neural Newtroks* (ANN), em inglês.

a fase de treinamento. A segunda emprega uma base de validação, que não faz parte da fase de treinamento, para configurar meta parâmetros dos classificadores e decidir em qual ponto está acontecendo o sobre-ajuste (ou *overfitting*, em inglês), ou seja, o ponto a partir do qual o modelo perde a capacidade de generalizar a função que define os dados e começa a descrever ruídos presentes na base de treino (JAMES et al., 2013). A base de teste continua com a mesma função que apresentava na primeira abordagem.

Visto que diversas abordagens de AM podem ser influenciadas pela ordem em que os dados são apresentados, é comum utilizar algum tipo de validação cruzada, técnica que separa aleatoriamente diferentes conjuntos de teste. Esse processo permite obter o resultado médio esperado de forma independente da ordem dos dados, uma vez que múltiplas combinações são utilizadas. O *k-fold* (em inglês) é um tipo de validação cruzada que separa os dados em k porções, utilizando uma porção como teste e as k *-1* restantes como treino, repetindo esse até ter utilizado todas as porções como treino (k vezes) (JAMES et al., 2013).

2.4.3 Árvores de decisão

Como o nome indica, Árvores de Decisão (ADs) são estruturas de árvores (binárias ou não) em que os nós-folhas contêm a classificação da amostra e todos os outros analisam intervalos (maior e menor que, igual, dentre outros) de um atributo. As ramificações correspondem às condições relacionadas ao valor do atributo, como, por exemplo, dia chuvoso, nublado ou ensolarado ((MITCHELL, 1997) e (JAMES et al., 2013)). A Figura 2.6, extraída de Mitchell (1997), mostra um exemplo de uma árvore de decisão.

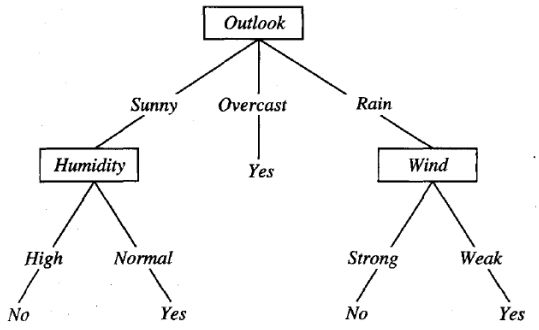


Figura 2.6: Exemplo de árvore de decisão não binária para o problema de decidir se alguém vai jogar tênis baseado na previsão do tempo (*Outlook*), umidade (*Humidity*) e vento (*Wind*). Os nós-folhas (nós-terminais) da árvore indicam a classe final. (Fonte: (MITCHELL, 1997)).

Diferentes algoritmos como o ID3 e as métricas de qualidade como o índice de *gini* podem ser empregados para definir os conjuntos e valores dos atributos a serem utilizados

pela árvore. Esse modelo apresenta as vantagens de não exigir um conjunto de atributos independentes entre si (PIROOZNIA et al., 2008) e ter fácil visualização e compreensão (JAMES et al., 2013).

Uma vez que uma árvore de decisão pode levar ao sobre-ajuste e alta variância entre diferentes treinamentos realizados para subconjuntos da mesma amostra de treino, é comum utilizar técnicas de reamostragem como, por exemplo, *bagging* e *boosting*. A primeira consiste em realizar n amostragens aleatórias com reposição (cada dado pode aparecer várias vezes em uma mesma amostra) da base de treinamento, treinar uma árvore para cada amostra e utilizar a média da predição entre as n árvores para classificação. A segunda consiste em expandir uma nova árvore com base na informação contida nas árvores existentes (JAMES et al., 2013).

O uso de *bagging* é uma das formas de *ensemble*, em inglês, e gera um modelo que possui um aglomerado de classificadores trabalhando em conjunto para rotular os dados (JAMES et al., 2013). Um conjunto de árvores de decisão que utiliza a técnica de *ensemble* pode ser chamado de floresta de decisão (HO, 1998).

Random Forest

Random forest é um modelo supervisionado baseado em floresta de decisão que, em geral, faz uso de *bagging*. Um dos algoritmos usados para construção de árvores de decisão para esse modelo é o ID3, que consiste em expandir os nós de cada uma das ADs utilizando o atributo que melhor separa a amostra de dados apresentada. Cada nova iteração do ID3 classifica parte do conjunto de dados iniciais, reduzindo o espaço amostral na fase de treinamento a cada iteração (MITCHELL, 1997).

O índice de *gini* (JAMES et al., 2013) mede a pureza dos dados - quanto menor, maior a predominância de uma das classes - e o ganho de informação (MITCHELL, 1997) é a redução da entropia do dado analisado ao selecionar um atributo para criar uma nova ramificação da árvore. Ambas são métricas quantitativas de impureza comumente empregadas para definir os atributos que melhor separam os dados. Como as variáveis que melhor realizam essa atividade estão em nós mais próximos à raiz, árvores menores são desejáveis, pois indicam menor grau de incerteza.

Devido ao fato de que algoritmos como o ID3 escolhem sempre os melhores atributos, é esperado que exista uma correlação entre as árvores criadas a partir do processo de *bagging*, aumentando a variância do modelo. Esse processo ocorre porque os melhores atributos que separam os dados tendem a ser os mesmos em todas as amostras. Para descorrelacionar as árvores resultantes, o modelo de *Random Forest* seleciona aleatoriamente um subconjunto de atributos a cada ramificação para realizar os cálculos de impureza. Dessa forma,

diferentes atributos tendem a ser utilizados a cada ramificação, reduzindo a correlação entre cada árvore e a variância do modelo (JAMES et al., 2013).

Isolation Forest

Isolation Forest (LIU; TING; ZHOU, 2008), em inglês, é um modelo não supervisionado baseado em 'floresta de decisão' que tem por objetivo detectar anomalias presentes em um conjunto de dados a partir dos atributos fornecidos.

Uma vez que, em geral, anomalias ocorrem em uma proporção bem menor do que os dados normais e, além disso, apresentam valores de atributos muito distintos, criar ramificações baseadas em valores aleatórios dos atributos (escolhidos entre os valores mínimo e máximo presentes na base de treino do atributo em questão) pode separar melhor as anomalias, já que tenderão a se localizar nos primeiros nós das árvores. A Figura 2.7, extraída de Liu, Ting e Zhou (2008), ilustra a separação de anomalias por partições aleatórias. A partir dela, pode-se perceber que as anomalias precisam de uma quantidade menor de ramificações para serem isoladas.

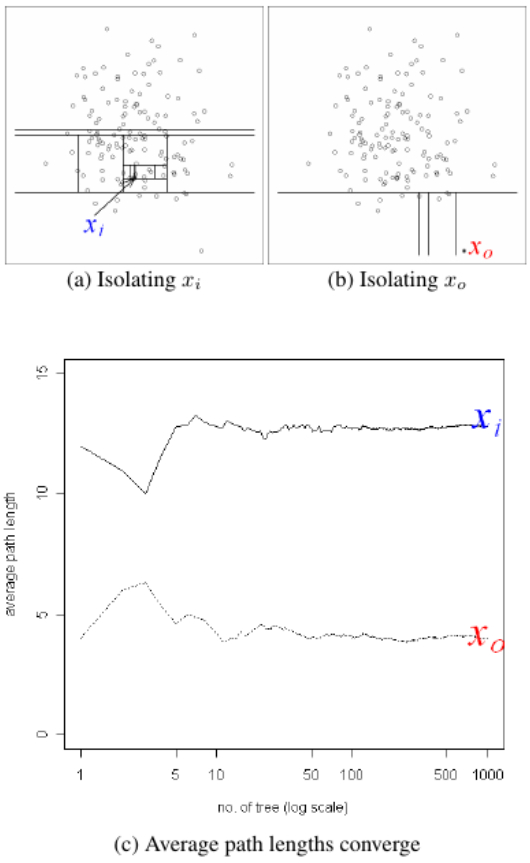
Após a criação das árvores, pode-se detectar novas anomalias utilizando uma proporção entre a esperança do número de vértices percorridos por uma entrada x e o número médio de vértices a se percorrer em um conjunto de dados de tamanho n. A partir dessa métrica, definida em Liu, Ting e Zhou (2008), pode-se concluir:

- se o número de vértices percorridos for próximo a 0, então a entrada fornecida é uma anomalia;
- caso contrário, a entrada pode ser considerada como normal.

2.5 Métricas de desempenho

Considerando que na aprendizagem supervisionada existem classes/rótulos predefinidos, então é esperado que o classificador consiga atribuir as classes esperadas para conjuntos de dados de entrada. A fim de avaliar o desempenho dos modelos na realização da tarefa de classificação, as métricas de revocação (*recall*, em inglês), precisão, acurácia e *f1-score* (em inglês) podem ser utilizadas.

Para o caso de duas classes, como, por exemplo, o problema de definir se uma entrada é ou não mutação, podem-se resumir as possíveis combinações de erros e acertos de acordo com a Tabela 2.1, chamada de matriz de confusão. Nela é possível ver as combinações das classes pré-definidas (linhas) e o resultado gerado pelo classificador (colunas), exemplificando os conceitos de Verdadeiro Positivo (classe de interesse que é predita corretamente), Falso Positivo (classe de interesse que não é predita corretamente), Verdadeiro Negativo



(a) Isolating x_i (b) Isolating x_o

(c) Average path lengths converge

Figura 2.7: Ilustração do processo de isolamento de anomalias. A imagem (a) mostra a quantidade de nós a serem percorridos para separar uma amostra normal (x_i), enquanto que a imagem (b) mostra essa mesma quantia para uma anomalia (x_o). A imagem (c) mostra o número médio de nós a se percorrer para isolar x_i e x_o à medida que o número de árvores cresce. (Fonte: (LIU; TING; ZHOU, 2008)).

(outras classes que são preditas corretamente), Falso Negativo (outras classes que não são preditas corretamente).

Tabela 2.1: Matriz de confusão para problema de 2 classes (é mutação x não é mutação). Indicadores de Verdadeiro Positivo, Falso Positivo, Falso Negativo e Verdadeiro Negativo para a classe é mutação.

	É mutação (classe predita)	Não é mutação (classe predita)
É mutação (classe esperada)	Verdadeiro Positivo	Falso Negativo
Não é mutação (classe esperada)	Falso Positivo	Verdadeiro Negativo

2.5.1 Revocação

Revocação é uma métrica calculada para cada classe, indicando a proporção de Verdadeiros Positivos recuperada dentre os dados realmente positivos, ou seja, Verdadeiro Positivo

(VP) e Falso Negativo (FN) para aquela classe. Pode ser calculada atráves da Equação 2.1.

$$\frac{VP}{VP + FN} \tag{2.1}$$

2.5.2 Precisão

Precisão é uma métrica calculada para cada classe, indicando a proporção de Verdadeiros Positivos corretamente classificada dentre os dados que o classificador julgou como 'positivo' para cada classe, ou seja, Verdadeiro Positivo (VP) e Falso Positivo (FP). Pode ser calculada através da Equação 2.2.

$$\frac{VP}{VP + FP} \tag{2.2}$$

2.5.3 Acurácia

Acurácia é uma métrica de acertos gerais do sistema. Indica a proporção de dados corretamente classificados dentre o total de dados na base. Não é indicada para bases de dados muito desbalanceadas, pois se um classificador atribuir a classe dominante para todas as entradas, ter-se-á acurácia elevada. Pode ser calculada através da Equação 2.3.

$$\frac{VP + VN}{VP + VN + FP + FN} \tag{2.3}$$

2.5.4 *F1-score*

F1-score é dada pela média harmônica entre precisão e revocação, contabilizando a atuação dessas métricas em conjunto. Pode ser calculada através da Equação 2.4.

$$\frac{2 * precisao * revocacao}{precisao + revocacao} \tag{2.4}$$

2.6 Detector de mutações somáticas utilizado

O detector de mutações somáticas chamado *Strelka2* (KIM et al., 2018) é capaz de indicar mutações germinativas e somáticas. Utiliza um modelo estocástico seguido da aplicação do algoritmo de *Random Forest* para permitir a melhora da precisão de acordo com a necessidade do usuário. A Figura 2.8, extraída de KIM et al. (2018), ilustra os procedimentos adotados para detecção de mutações germinativas (imagem 'a') e somáticas (imagem 'b').

É importante ressaltar que, como mostra a Figura 2.8 (imagem 'b'), o modelo estocástico do *Strelka2* (KIM et al., 2018) faz uma pré-seleção das mutações somáticas antes da etapa de aplicação do *Random Forest*. Além disso, calcula os atributos que serão fornecidos como entrada para o modelo de AM que serve para reavaliar se as mutações somáticas indicadas na fase anterior são realmente mutações. Esse processo pode acarretar na perda de algumas mutações na fase anterior ao uso de AM, mas isso será ignorado no presente trabalho, uma vez que se deseja comparar a performance do *Isolation Forest* com o *Random Forest* utilizando o *Strelka2* (KIM et al., 2018).

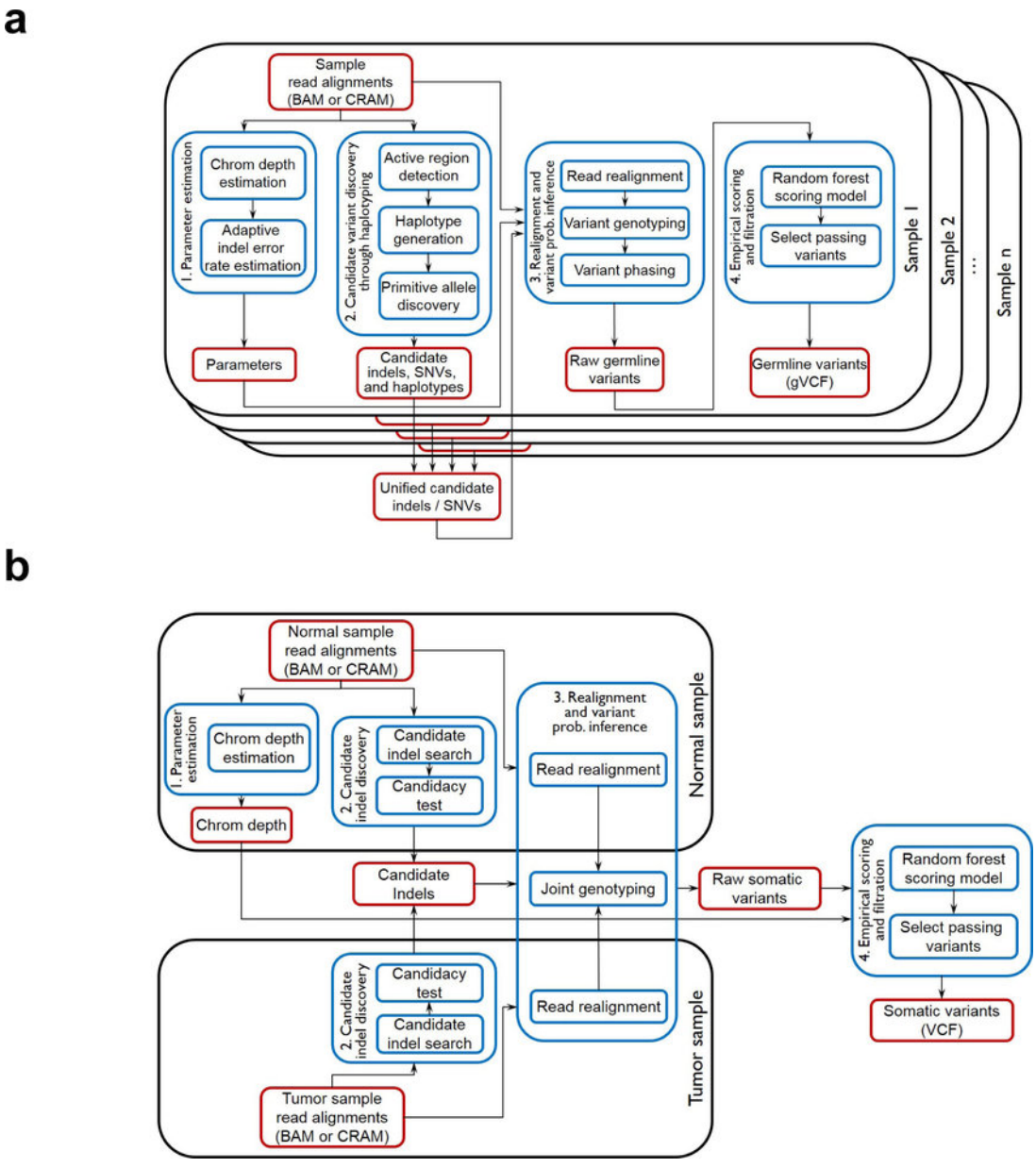


Figura 2.8: Procedimentos para detectar mutações germinativas ('a') e somáticas ('b') adotados pelo *Strelka2*. (Fonte: (KIM et al., 2018)).

2.7 Revisão da literatura

Considerando a questão de detecção de mutações somáticas, diversas abordagens foram empregadas na literatura, incluindo modelos estocásticos ((KIM et al., 2018), (LAI et al., 2016) e (FAN et al., 2016)), puramente baseados em AM (SAHRAEIAN et al., 2019) e uso de combinação de atributos de diversos classificadores como entradas para modelos de AM (ANZAR et al., 2019).

Dentro dos modelos estocásticos, o *Strelka2* (KIM et al., 2018) e o *Muse* (FAN et al., 2016) utilizam como base a frequência de alelos para analisar as probabilidades (XU, 2018). O *Vardict* (LAI et al., 2016) tem uma abordagem diferente, combinando a heurística com testes estatísticos para identificar as mutações (XU, 2018). É importante lembrar que o *Strelka2* (KIM et al., 2018) combina o modelo estocástico (primeira fase) com AM (segunda fase), utilizando o *Random Forest* para aumentar a precisão de detecção das mutações obtidas na primeira fase (processo ilustrado na Figura 2.8).

Dentre os classificadores que se baseiam puramente em AM, sem utilizarem um modelo estocástico para realizar uma pré-seleção das regiões e atributos, o *Neusomatic* (SAHRAEIAN et al., 2019) usa redes neuronais convolucionais (LECUN et al., 1990) do tipo *ResNet* (HE et al., 2016) para automaticamente extrair atributos importantes para a detecção de mutações somáticas a partir dos dados brutos dos arquivos de alinhamento (BAM). Essa abordagem permitiu que o método de Sahraeian et al. (2019) conseguisse, por exemplo, ser melhor ou igual aos demais modelos avaliados para o experimento de duas misturas da base *Platinum*, atingindo um *f1-score* de 0,995 para a mistura 70% tumor e 95% normal, enquanto que o *Strelka2* (KIM et al., 2018), o *Vardict* (LAI et al., 2016) e o *Muse* (FAN et al., 2016) obtiveram desempenho de, respectivamente, 0,959, 0,815 e 0,635.

Outra maneira de aumentar a qualidade da detecção de mutações somáticas é combinar atributos de diversos classificadores existentes (como os estocásticos, por exemplo) e utilizar esses atributos como entradas para modelos de AM, como feito pelo *NeoMutate* (ANZAR et al., 2019). Os autores do *NeoMutate* se basearam no princípio de que cada classificador utilizado em separado era muito específico para um determinado cenário de mutação, sendo muito bom para ele e não tão bom para os demais. Devido a essa alta especificidade, a união de diversos classificadores em um sistema de AM poderia compensar as falhas de cada classificador tomado individualmente.

Capítulo 3

Materiais e Métodos

Os métodos e as configurações dos dispositivos utilizados para comparar os desempenhos obtidos pelos modelos de AM *Random Forest* e *Isolation Forest* estão descritos a seguir.

3.1 Bases de dados

Foram baixados do servidor *amazonAWS*[1], descrito nas notas complementares do *preprint*, em inglês, do *Strelka2* (KIM et al., 2018), os seguintes arquivos:

- os sequenciamentos públicos de amostras de DNA de um paciente com doença genética (*NA12878*, servindo como a amostra de 'tumor'/doente) e outro sem doença conhecida (*NA12877*, servindo como a amostra 'normal');
- os arquivos de alinhamento BAM para os experimentos de mistura de pureza tumor-normal *T80-N90* e *T80-N100*;
- os arquivos de sequenciamento de referência no formato FASTA (*GRCh38Decoy*);
- os arquivos contendo as mutações somáticas analisadas (VCFs e BEDs para *InSilicoMix_indels* e *InSilicoMix_SNVs*);
- os arquivos de índice (FAI, BAI e TBI);
- o arquivo *callable.bed* (conforme as recomendações fornecidas no material complementar do *Strelka2* (KIM et al., 2018)).

Assim como feito pelos autores do *Strelka2* (KIM et al., 2018), os cromossomos 2 e 20 foram utilizados como base de testes, enquanto o restante, como base de treino.

[1]Link para acesso aos dados utilizados *<https://s3.amazonaws.com/strelka-public/>*. Acessado em 12/10/2019

O servidor da *amazonAWS* (referenciado no *pre-print*) foi empregado em detrimento do servidor da NCBI (referenciado na versão final do artigo de Kim et al. (2018))[2] devido ao fato de que os arquivos do NCBI foram salvos em um formato próprio, exigindo, por isso, um tempo significativo para conversão para os formatos utilizados pelo *Strelka2*. Uma vez que os arquivos no servidor *amazonAWS* estavam imediatamente disponíveis após serem baixados, optou-se por eles.

A mistura tumor normal *T20-N100* não foi utilizada porque problemas de conexão ocorreram nas tentativas realizadas de baixar os arquivos necessários. Como os arquivos BAM eram da ordem de dezenas a centenas de GB, até mesmo a retomada da operação de obtenção dos arquivos levaria um tempo muito grande com a conexão instável. Como o objetivo do projeto é comparar o desempenho do modelo *Isolation Forest* com o *Random Forest*, a não utilização dessa base não causou impacto significativo para a análise.

Visto que os dados mais discrepantes com o menor percentual na base de treino são considerados pelo *Isolation Forest* como amostras anormais, calculou-se o percentual das amostras que eram mutação e não mutação para as bases de teste e treinamento, considerando SNVs e *indels* nas bases utilizadas (*T80-N90* e *T80-N100*). As Tabelas 3.1 a 3.2 contêm essas distribuições entre classes.

Tabela 3.1: Distribuição dos dados entre as classes (mutação x não mutação) de treino e teste para SNVs e *indels* considerando a base de dados *T80-N90*.

	Experimento: *T80-N90*	
	Mutação	Não Mutação
SNV	27,45% (treino) | 28,14% (teste)	72,55% (treino) | 71,86% (teste)
indels	32,33% (treino) | 33,60% (teste)	67,67% (treino) | 66,40% (teste)

Tabela 3.2: Distribuição dos dados entre as classes (mutação x não mutação) de treino e teste para SNVs e *indels* considerando a base de dados *T80-N100*.

	Experimento: *T80-N100*	
	Mutação	Não Mutação
SNV	26,09% (treino) | 26,70% (teste)	73,91% (treino) | 73,30% (teste)
indels	29,08% (treino) | 29,92% (teste)	70,92% (treino) | 70,08% (teste)

[2]Link para acesso aos dados utilizados no servidor do NCBI referenciado na versão final do Strelka2 (KIM et al., 2018) <*https://www.ncbi.nlm.nih.gov/sra/SRP142632*>. Acessado em 12/10/2019

3.2 Linguagem de programação utilizada

A linguagem de programação *Python* ™ foi utilizada para a primeira etapa de filtragem e obtenção de atributos realizada pelo detector de mutações somáticas chamado *Strelka2* (KIM et al., 2018) e, também, para treinar e avaliar o desempenho dos modelos de AM. Essa escolha foi motivada pelo fato de que o *Strelka2* (KIM et al., 2018) implementa as rotinas de treino de modelos AM em *Python* ™ e também pela praticidade e versatilidade em se desenvolver aplicações nessa linguagem. As versões 2.7.12 para *Python2* e 3.5.2 para *Python3* foram empregadas. Essa última foi utilizada para parte de AM por ter um tempo de execução menor, enquanto que a primeira executou a primeira etapa de filtragem do *Strelka2* (KIM et al., 2018).

A biblioteca *Scikit-learn* (PEDREGOSA et al., 2011), implementada para a linguagem *Python* ™, foi usada a fim de facilitar o desenvolvimento de diversas aplicações de AM (BUITINCK et al., 2013) como, por exemplo, ADs. A versão 0.21.3 foi utilizada para treinar os modelos *Isolation Forest* e *Random Forest* no *Python3*.

3.3 Configuração do *Hardware*

A fim de treinar e avaliar os modelos de AM, bem como realizar toda etapa de filtragem e cálculo de atributos presentes na implementação do *Strelka2* (KIM et al., 2018), um *notebook* com Intel Core i7-5500U (2 núcleos físicos), 250 GB de memória (SSD) e 8 GB de RAM foi utilizado para avaliar *indels*, enquanto que o processamento de SNVs foi feito em um computador de mesa com Intel Core i7-8700 (6 núcleos físicos), 250 GB de memória (SSD) e 32 GB de RAM. Máquinas diferentes foram utilizadas para cada tipo de mutação porque o número de SNVs era, aproximadamente, 10x maior que a quantidade de *indels*.

Para armazenar os arquivos de entrada (alinhamentos, sequenciamentos e posições das mutações anotadas), além de fornecer uma memória *swap* de 100GB para processos que necessitavam de mais de RAM, utilizou-se um HD externo de 2TB.

3.4 Configuração do *Strelka2* utilizada

Para executar a filtragem estocástica e calcular os atributos, criando os *Raw somatic variants* descritos na Figura 2.8, o *Python2* (versão 2.7.12) foi usado com as configurações padrão pré compiladas do *Strelka2* (KIM et al., 2018). Os autores do *Strelka2* (KIM et al., 2018) sugerem a utilização do Manta (CHEN et al., 2015) em conjunto com o *Strelka2* (KIM et al., 2018) para melhorar a detecção de *indels*. A combinação deles exigiria

um tempo de execução muito grande por arquivo. Visto que o uso de cada uma dessas ferramentas é muito custoso em termos de tempo e que os autores do *Strelka2* (KIM et al., 2018) mencionaram que as bases de dados utilizadas não apresentavam resultados muito diferentes caso o Manta fosse usado[3], somente o *Strelka2*, sem o uso do Manta, foi executado para a primeira etapa. A versão mais atual do *Strelka2* (KIM et al., 2018) até o momento, 2.9.10 (centos6_x86_64)[4], foi utilizada no Ubuntu 16.04 LTS (*notebook*) e 18.04.2 LTS (outro computador).

Após a primeira etapa de filtragem, os atributos a serem utilizados para as mutações somáticas estavam disponíveis para serem usados pelos modelos de AM. Esses atributos, extraídos do material complementar de Kim et al. (2018), estão descritos abaixo:

- *SomaticSNVQualityAndHomRefGermlineGenotype* e *SomaticIndelQualityAndHomRefGermlineGenotype*: probabilidade a posteriori de mutações do tipo SNVs ou *indels* condicionadas em um genótipo germinativo homozigótico de referência;
- *NormalSampleRelativeTotalLocusDepth*: profundidade relativa do *locus* com relação à esperança, dada pela razão da profundidade total da leitura no *locus* variante, incluindo qualquer qualidade de mapeamento, na amostra normal;
- *TumorSampleAltAlleleFraction*: Fração das observações da amostra de tumor que não estão no alelo de referência. Limitado a um máximo de 0.5;
- *RMSMappingQuality*: raiz quadrática média da qualidade das leituras de todas aquelas ao longo da variante em todas as amostras;
- *ZeroMappingQualityFraction*: Fração das qualidades de mapeamento da leitura iguais a 0;
- *InterruptedHomopolymerLength*: tamanho do maior homopolímero interrompido menos um;
- *TumorSampleStrandBias*: razão logarítmica da probabilidade do alelo somático do tumor assumindo que ele ocorra somente em uma fita de DNA versus que ocorra em ambas;
- *TumorSampleReadPosRankSum*: valor z do teste U de Mann-Whitney para a referência contra a não referência das posições de leitura de alelos nas observações de amostras de tumor;

[3]Link para comentário de um dos autores do *Strelka2* (KIM et al., 2018) (Sangtae Kim) sobre o uso do Manta (CHEN et al., 2015) nas bases de dados utilizadas por eles *<https://github.com/Illumina/strelka/issues/72>*. Acessado em 13/10/2019.

[4]Link para acesso às diferentes versões do *Strelka2* (KIM et al., 2018). *<https://github.com/Illumina/strelka/releases>*. Acessado em 13/10/20198.

- *AlleleCountLogOddsRatio*: as razões de probabilidade, em escala logarítmica, da contagem de alelos $log\frac{r_t a_n}{r_n a_t}$, dadas as contagens de alelos da referência (r_t, r_n) e não referência (a_t, a_n) do par de amostras (tumor e normal);
- *NormalSampleFilteredDepthFraction* e *TumorSampleFilteredDepthFraction*: fração de leituras que foram filtradas da amostra (tumor ou normal) antes de detectar o *locus* variante;
- *TumorSampleLogSymmetricStrandOddsRatio*: Logarítmo da razão das probabilidades de fita de DNA simétricas de contagem de alelos $log(\frac{r_{fwd}a_{rev}}{r_{rev}a_{fwd}} + \frac{r_{rev}a_{fwd}}{r_{fwd}a_{rev}})$ dadas as contagens de confiança das observações da amostra de tumor, considerando a referência (r_{fwd}, r_{rev}) e não referência (a_{fwd},a_{rev});
- *RepeatUnitLength*: tamanho da unidade de repetição do alelo de *indel* somático;
- *IndelRepeatCount* e *RefRepeatCount*: número de vezes que a unidade de repetição do alelo de *indel* somático ocorre na sequência de referência (*RefRepeatCount*) e em um haplótipo contendo o alelo de *indel* (*IndelRepeatCount*);
- *TumorSampleIndelNoiseLogOdds* e *TumorNormalIndelAlleleLogOdds*: razão logarítmica da frequência do *indel* candidato contra todos os outros *indels* no mesmo *locus* da amostra de tumor (*TumorSampleIndelNoiseLogOdd*) ou contra as amostras normais (*TumorNormalIndelAlleleLogOdd*).

A Tabela 3.3 mostra quais deles são usados para SNVs e quais são *indels*.

Tabela 3.3: Atributos utilizados pelo *Strelka2* (KIM et al., 2018) e tipo de mutação onde cada atributo é utilizado (SNV ou *indels*).

Nome atributo	Usado em
SomaticSNVQualityAndHomRefGermlineGenotype	SNV
NormalSampleRelativeTotalLocusDepth	SNV
TumorSampleAltAlleleFraction	SNV
RMSMappingQuality	SNV
ZeroMappingQualityFraction	SNV
TumorSampleStrandBias	SNV
NormalSampleFilteredDepthFraction	SNV
TumorSampleFilteredDepthFraction	SNV
TumorSampleReadPosRankSum	SNV e *indels*
AlleleCountLogOddsRatio	SNV e *indels*
SomaticIndelQualityAndHomRefGermlineGenotype	*indels*
TumorSampleLogSymmetricStrandOddsRatio	*indels*
IndelRepeatCount	*indels*
InterruptedHomopolymerLength	*indels*
RefRepeatCount	*indels*
RepeatUnitLength	*indels*
TumorSampleIndelNoiseLogOdds	*indels*
TumorNormalIndelAlleleLogOdds	*indels*

3.5 Parâmetros utilizados para o *Isolation Forest*

Dados os atributos para SNVs e *indels* resumidos na Tabela 3.3, comparou-se o desempenho do *Isolation Forest* com relação ao *Random Forest*. O primeiro teve os seguintes parâmetros testados: contaminação, quantidade de atributos para o treinamento, tamanho do conjunto de amostragem para treinar cada árvore e número de árvores (*contamination*, *max_features*, *max_samples* e *n_estimators* respectivamente), enquanto que o segundo permaneceu com as configurações utilizadas pelos autores do *Strelka2* (KIM et al., 2018) (*max_depth*=6, *n_estimators*=100, tendo os demais mantidos com os valores padrão do *Scikit-learn* (PEDREGOSA et al., 2011)).

Cada uma dessas variáveis utilizadas no *Isolation Forest* foi avaliada em 10 valores diferentes, mantendo-se as demais em um valor constante, a fim de quantificar o desempenho que cada variável tem separadamente sobre a qualidade da detecção de mutações somáticas. Os valores utilizados em cada teste, bem como a ordem deles são descritos a seguir. Para obter resultados estáveis independente da ordem dos dados de treinamento, uma validação cruzada do tipo *k-fold* (k=10) foi feita somente 1 vez para cada combinação de base, parâmetro e tipo mutação. Cada parâmetro foi avaliado com base no valor médio obtido para as métricas de desempenho e os classificadores, cujos resultados eram mais próximos à média, foram utilizados na fase de testes.

Visando mitigar possíveis influências da ordem das amostras nas bases de testes, dividiram-se ambas em porções de $\frac{1}{3}$ e $\frac{2}{3}$ e se avaliou se os resultados eram semelhantes entre cada conjunto. Compararam-se os resultados obtidos após essas otimizações dos parâmetros com relação às configurações padrões do *Isolation Forest* considerando a descrição da versão 0.22 presente para o *Scikit-learn* (PEDREGOSA et al., 2011) (*contamination*='auto', *max_samples*='auto', *n_estimators*=100 e *behaviour*='new'), a fim de quantificar os ganhos obtidos pela busca de melhores parâmetros desse modelo.

3.5.1 Variável: contaminação (*contamination*)

Visto que a contaminação indica para o modelo de *Isolation Forest* do *Scikit-learn* (PEDREGOSA et al., 2011) o percentual de anomalias presentes na base de dados, optou-se por visualizar o desempenho obtido ao utilizar valores entre 0.05 e 0.5, cada um espaçado 0.05 entre si. Como evidenciam as Tabelas 3.1 a 3.2, as amostras de Mutação apresentam as menores distribuições nas bases de treino e teste, sendo assim consideradas como as anomalias ou 'contaminação'. Para as etapas subsequentes, escolheu-se o valor de contaminação que apresentava maior revocação da classe Mutação, mantendo essa métrica maior ou igual a 0.6 para a classe Não Mutação, uma vez que a primeira era diretamente proporcional à contaminação, enquanto que a segunda era inversamente proporcional.

3.5.2 Variável: número máximo de atributos (*max_features*)

Com a contaminação definida, o número de atributos utilizados por árvore foi avaliado. Devido ao fato de que somente 10 atributos são utilizados para SNVs e *indels* (respectivamente aqueles compreendidos de *SomaticSNVQualityAndHomRefGermlineGenotype* a *AlleleCountLogOddsRatio* e entre *TumorSampleReadPosRankSum* e *TumorNormalIndelAlleleLogOdds* na Tabela 3.3), foi viável analisar todos os valores possíveis para essa variável ao comparar o desempenho utilizando entre 1 e 10 atributos. Escolheu-se o valor que resultava em uma das melhores revocações para a classe de mutações.

3.5.3 Variável: número máximo de amostras (*max_samples*)

Após definir a contaminação e o número de atributos a se utilizar, investigou-se o impacto da subamostragem (*max_samples*) durante a fase de treinamento. Seguindo a escala exponencial utilizada em Liu, Ting e Zhou (2008) para avaliar esse parâmetro, os seguintes valores foram utilizados: 2, 4, 8, 16, 32, 128, 256, 512, 2048 e 8192 amostras. O valor que resultava uma das melhores revocações para a classe de mutações foi escolhido.

3.5.4 Variável: total de árvores utilizadas (*n_estimators*)

Por fim, o número de árvores, ou estimadores, utilizados para a classificação foi explorado. A escolha de *n_estimators* como último parâmetro a ser avaliado se deu devido ao fato de que o uso desse parâmetro em estágios iniciais poderia levar a um número de estimadores muito baixo devido à pequena presença de 'anomalias' (mutações) nas bases de treino. Conjuntos de 1, 5, 10, 30, 50, 100, 150, 200, 250 e 300 árvores de decisões foram avaliados.

A Tabela 3.4 resume os parâmetros avaliados para os experimentos principais.

Tabela 3.4: Resumo dos parâmetros utilizados para os experimentos principais.

Ordem Experimento	Nome atributo	Valores avaliados
1	*contamination*	0.05, 0.1, 0.15, 0.2, 0.25, 0.3, 0.35, 0.4, 0.45, 0.5
2	*max_features*	1, 2, 3, 4, 5, 6, 7, 8, 9, 10
3	*max_samples*	2, 4, 8, 16, 32, 128, 256, 512, 2048, 8192
4	*n_estimators*	1, 5, 10, 30, 50, 100, 150, 200, 250, 300

3.5.5 Testes adicionais

Como havia a possibilidade de se obter resultados melhores na faixa de 50 a 100 estimadores, uma avaliação desse intervalo foi realizada. Foram tomados 11 pontos separados em um passo de 5 estimadores, incluindo os extremos. A contaminação, número de atributos e número de amostras foram deixados nas melhores configurações obtidas com os

experimentos anteriores. Todos os demais parâmetros do *Isolation Forest* foram deixados em seus valores predefinidos pelo *Scikit-learn* (PEDREGOSA et al., 2011).

Além dos procedimentos descritos, níveis de contaminação de 0.5, 'auto' (conforme a documentação do *Scikit-learn* (PEDREGOSA et al., 2011)) e os valores das distribuições de mutação presentes na Tabela 3.1 foram utilizados a fim de analisar a possibilidade de melhoria do desempenho após definir valores para os demais parâmetros. De forma análoga à segunda avaliação do número de árvores, todos os demais parâmetros foram deixados nas melhores configurações obtidas.

Capítulo 4

Resultados e Discussão

Todos os testes definidos no Capítulo 3 foram executados para os tipos de variações analisadas (SNVs e *indels*) em ambas as bases de dados utilizadas (*T80_N90* e *T80_N100*). Graças ao uso da validação cruzada, que possibilitou extrair o desempenho médio para cada parâmetro, os resultados apresentados na fase de treino e teste (aproximadamente $\frac{1}{3}$ e $\frac{2}{3}$ de cada base) não foram muito diferentes entre si. O emprego da validação cruzada foi necessário, uma vez que, inicialmente, realizaram-se os testes de parâmetros removendo a aleatoriedade da ordem em que os dados eram exibidos para o *Isolation Forest* na fase de treino. Esse processo gerou resultados muito instáveis, com grandes variações ocorrendo se quaisquer outra ordem dos dados fosse utilizada. Com o uso do *k-fold*, os resultados obtidos apresentaram menor variação entre si, mantendo-se próximos entre si nas fases de validação e teste.

Assim, somente aqueles obtidos para a divisão em $\frac{2}{3}$ das bases de testes serão utilizados. Essa fração foi escolhida por conter a maior porção de dados da última fase, permitindo um melhor indicativo da performance geral do *Isolation Forest*. Para cada parâmetro, serão exibidas as métricas de revocação (Equação 2.1) e *f1-score* (Equação 2.4), uma vez que esta última contém uma média entre a precisão (Equação 2.2) e a revocação.

Por questão de nomenclatura adotada pelos autores do *Strelka2* (KIM et al., 2018) ao disponibilizar os códigos para treinamento de ADs após a fase inicial de filtragem, a classe mutação foi chamada de *TP* (equivalente a obter um VP para a classe de mutação) e a classe não mutação foi chamada de *FP* (uma vez que, idealmente, o filtro inicial do *Strelka2* (KIM et al., 2018) deveria permitir somente mutações. Por isso, todas as amostras que não são mutações e estão na etapa de AM são consideradas como FP).

Visto que o objetivo do trabalho é avaliar a performance de métodos de AM após a filtragem realizada pelo *Strelka2* (KIM et al., 2018), todas aquelas mutações que foram descartadas nessa primeira fase do *Strelka2* (KIM et al., 2018) (Falso Negativo) não foram contabilizadas nos resultados.

Após a apresentação e definição dos melhores parâmetros para esse modelo, uma comparação do resultado obtido por ele e pelo *Random Forest* será realizada. Em seguida os resultados serão discutidos, ressaltando-se as partes mais importantes.

4.1 *Indels*

O primeiro tipo de mutação a ser investigado foram *indels*. A ordem dos parâmetros, bem como os valores nos testes iniciais, segue de acordo com o definido na Tabela 3.4.

4.1.1 Variável: contaminação (*contamination*)

A contaminação para os *indels* apresentou comportamento diretamente proporcional com relação às mutações (classe *TP*) e inversamente proporcional com relação a dados que não constituem mutações de fato (classe *FP*). As Figuras 4.1 a 4.2 exemplificam esse comportamento.

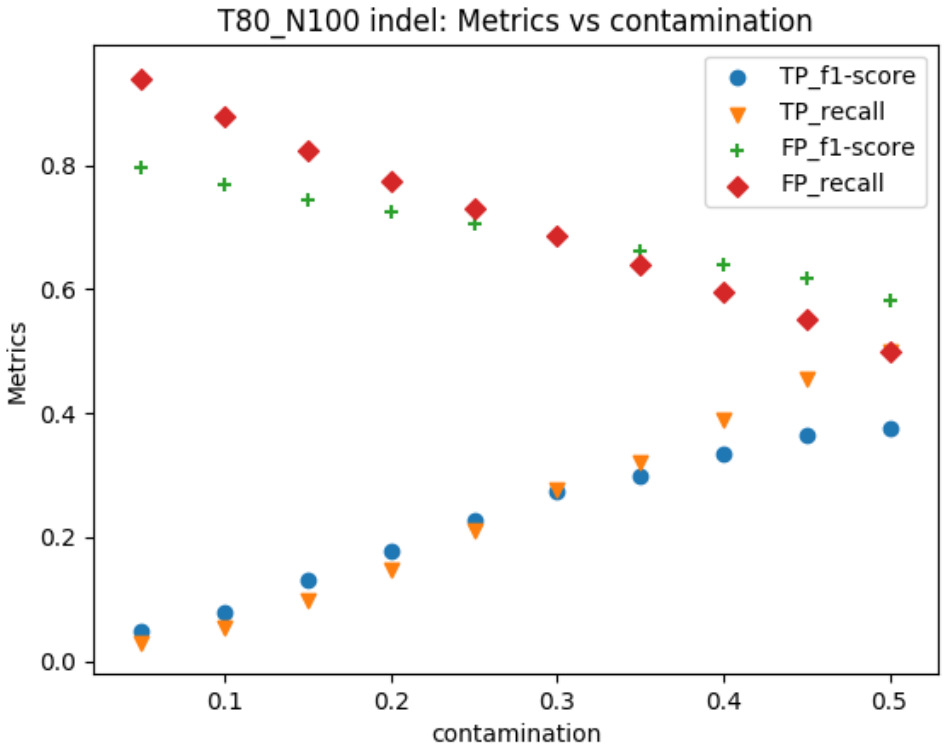


Figura 4.1: Influência da contaminação, considerando os *indels* presentes na base *T80_N100*, nas métricas de revocação da classe mutação (*TP_recall*, triângulo em laranja), revocação da classe não mutação (*FP_recall*, diamante em vermelho), *f1-score* da classe mutação (*TP_f1-score*, círculo em azul) e *f1-score* da classe não mutação (*FP_f1-score*, sinal de adição em verde).

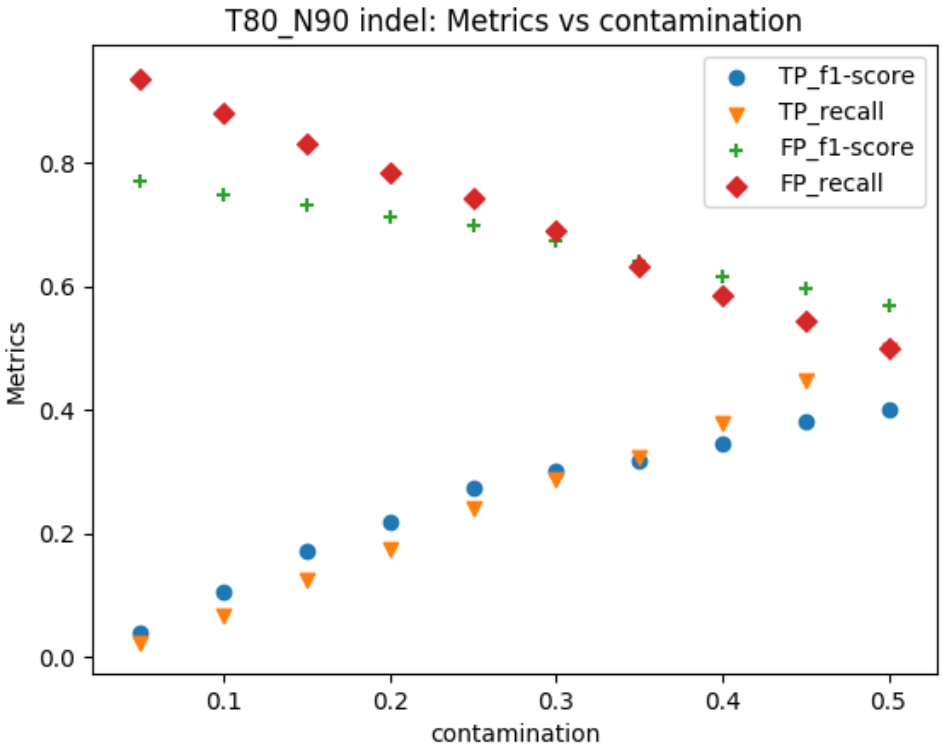


Figura 4.2: Influência da contaminação, considerando os *indels* presentes na base *T80_N90*, nas métricas de revocação da classe mutação (*TP_recall*, triângulo em laranja), revocação da classe não mutação (*FP_recall*, diamante em vermelho), *f1-score* da classe mutação (*TP_f1-score*, círculo em azul) e *f1-score* da classe não mutação (*FP_f1-score*, sinal de adição em verde).

Como esse equilíbrio entre as classes pode não gerar ganhos significativos, escolheu-se o valor de contaminação que resultasse na maior revocação de *TP*, mantendo essa métrica maior ou igual a 0.6 para *FP*. Sendo assim, uma contaminação de 0.35 foi escolhida para as etapas subsequentes.

4.1.2 Variável: número máximo de atributos (*max_features*)

Seguindo a ordem estabelecida, o número de atributos a serem considerados para construir cada estimador foi avaliado. Uma fração, com relação ao máximo, de atributos foi definida. Devido ao fato de que o *Strelka2* (KIM et al., 2018) utiliza 10 atributos para *indels*, o parâmetro (*max_features*) cobriu todas as possbilidades. Como mostram as Figuras 4.3 a 4.4, utilizar somente um atributo por árvore gerou os melhores resultados (em termos de revocação e *f1-score*) para a detecção das mutações (*TP*). Sendo assim, $max\_features = 0.1$ foi definido para as próximas etapas.

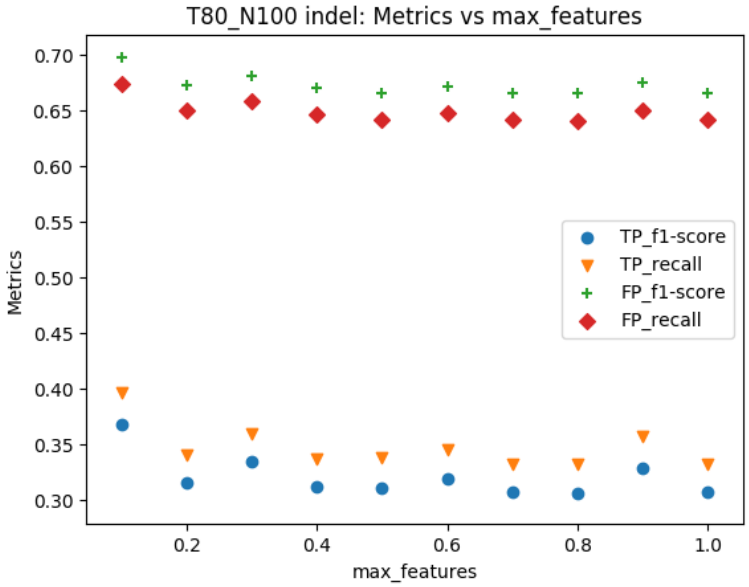


Figura 4.3: Influência do número máximo de atributos, considerando os *indels* presentes na base *T80_N100*, nas métricas de revocação da classe mutação (*TP_recall*, triângulo em laranja), revocação da classe não mutação (*FP_recall*, diamante em vermelho), *f1-score* da classe mutação (*TP_f1-score*, círculo em azul) e *f1-score* da classe não mutação (*FP_f1-score*, sinal de adição em verde).

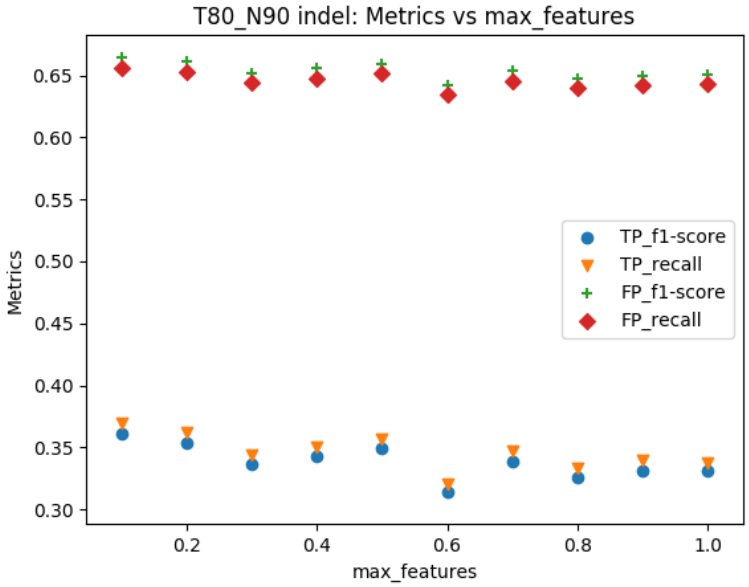


Figura 4.4: Influência do número máximo de atributos, considerando os *indels* presentes na base *T80_N90*, nas métricas de revocação da classe mutação (*TP_recall*, triângulo em laranja), revocação da classe não mutação (*FP_recall*, diamante em vermelho), *f1-score* da classe mutação (*TP_f1-score*, círculo em azul) e *f1-score* da classe não mutação (*FP_f1-score*, sinal de adição em verde).

4.1.3 Variável: número máximo de amostras (*max_samples*)

Empregando-se uma contaminação de 0.35 e 1 atributo para a formação de cada árvore ($max\_features = 0.1$), o tamanho da amostra aplicada para treinar cada uma delas foi utilizado. Os dados presentes nas Figuras 4.5 a 4.6 sugerem um valor máximo das métricas de mutação (*TP*) para 8 amostras. Em ambas as bases de dados, 2 amostras resultaram nas menores métricas da classe *TP*. A base *T80_N90* (Figura 4.6) apresenta métricas semelhantes para a classe de mutações para os valores de 8 e 128 de *max_samples* enquanto que a base *T80_N100* (Figura 4.5) apresentou melhor desempenho somente com $max\_samples = 8$. Por uma questão de tornar os parâmetros independentes da base, escolheu-se $max\_samples = 8$ para as próximas etapas.

Devido à organização dos parâmetros de modo exponencial, a escala mono-log (eixo dos parâmetros em escala logarítmica e eixo das métricas em escala linear) foi utilizada para facilitar a compreensão dos resultados.

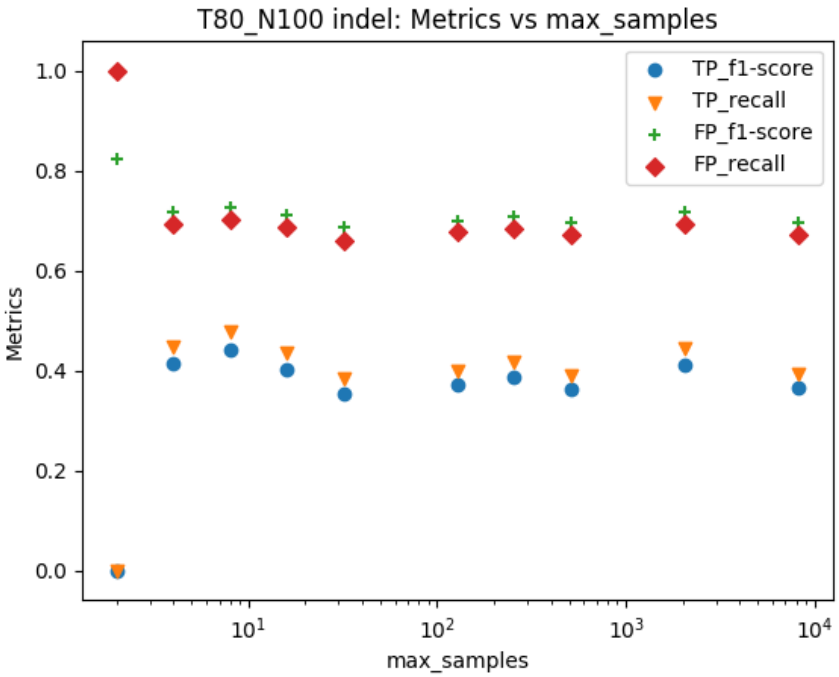


Figura 4.5: Influência do tamanho máximo da amostragem, considerando os *indels* presentes na base *T80_N100* nas métricas de revocação da classe mutação (*TP_recall*, triângulo em laranja), revocação da classe não mutação (*FP_recall*, diamante em vermelho), *f1-score* da classe mutação (*TP_f1-score*, círculo em azul) e *f1-score* da classe não mutação (*FP_f1-score*, sinal de adição em verde).

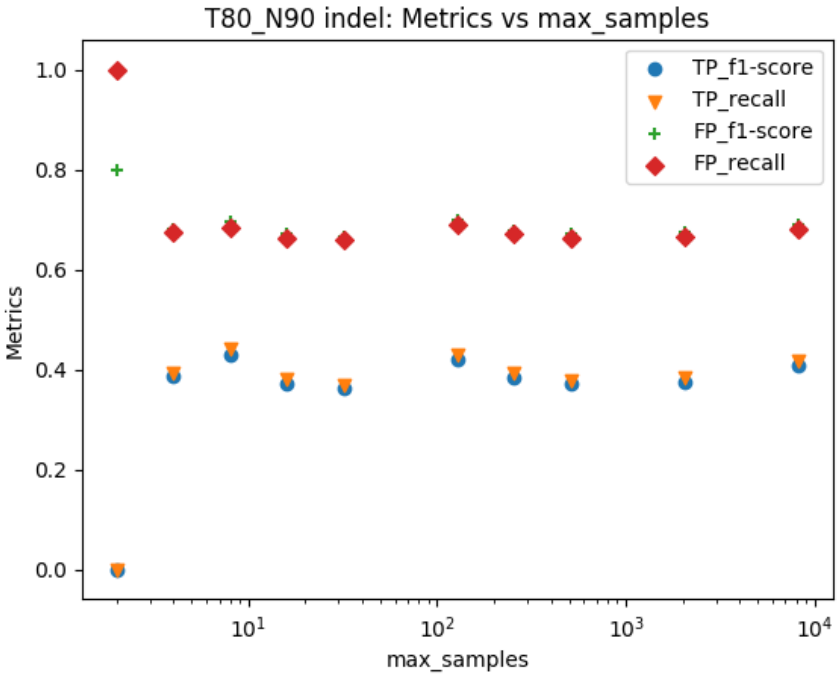


Figura 4.6: Influência do tamanho máximo da amostragem, considerando os *indels* presentes na base *T80_N90* nas métricas de revocação da classe mutação (*TP_recall*, triângulo em laranja), revocação da classe não mutação (*FP_recall*, diamante em vermelho), *f1-score* da classe mutação (*TP_f1-score*, círculo em azul) e *f1-score* da classe não mutação (*FP_f1-score*, sinal de adição em verde).

4.1.4 Variável: total de árvores utilizadas (*n_estimators*)

Para finalizar a análise inicial de parâmetros, o total de árvores utilizado para detectar mutações foi avaliado. Visto que os melhores números de atributos e amostras por árvore foram baixos (1 e 8 respectivamente), a quantidade de estimadores utilizados se torna a maior responsável pelo tempo de execução em termos de treinamento e predição. Assim, a melhor configuração possível é aquela que tem as melhores métricas de desempenho utilizando o menor número de ADs.

É possível observar nas Figuras 4.7 a 4.8 que as melhores revocações e *f1-scores* estão na faixa entre 50 e 100 ADs. Para a base *T80_N100*, os melhores resultados foram obtidos com $n\_estimators = 100$, enquanto que para o outro conjunto de dados (*T80_N90*), o melhor valor desse parâmetro foi 50. Uma vez que esses valores, para ambas as bases, estão próximos a regiões de máximos para a classe de mutações (*TP*), optou-se por investigar, na Seção 4.1.5, o comportamento da variável *n_estimators* nesse intervalo. Desse modo, espera-se que exista um mesmo valor que dê os melhores resultados para ambos *T80_N90* e *T80_N100*.

Devido à realização de uma nova busca da quantidade de ADs dentro de um intervalo

menor (de 50 a 100 árvores), não se escolheu valor para o parâmetro *n_estimators* nesta etapa. Assim, o primeiro teste adicional para *indels* terá as seguintes variáveis configuradas: *contamination*=0.35, *max_features*=0.1 (1 atributo por árvore) e *max_samples*=8 (amostras por estimador).

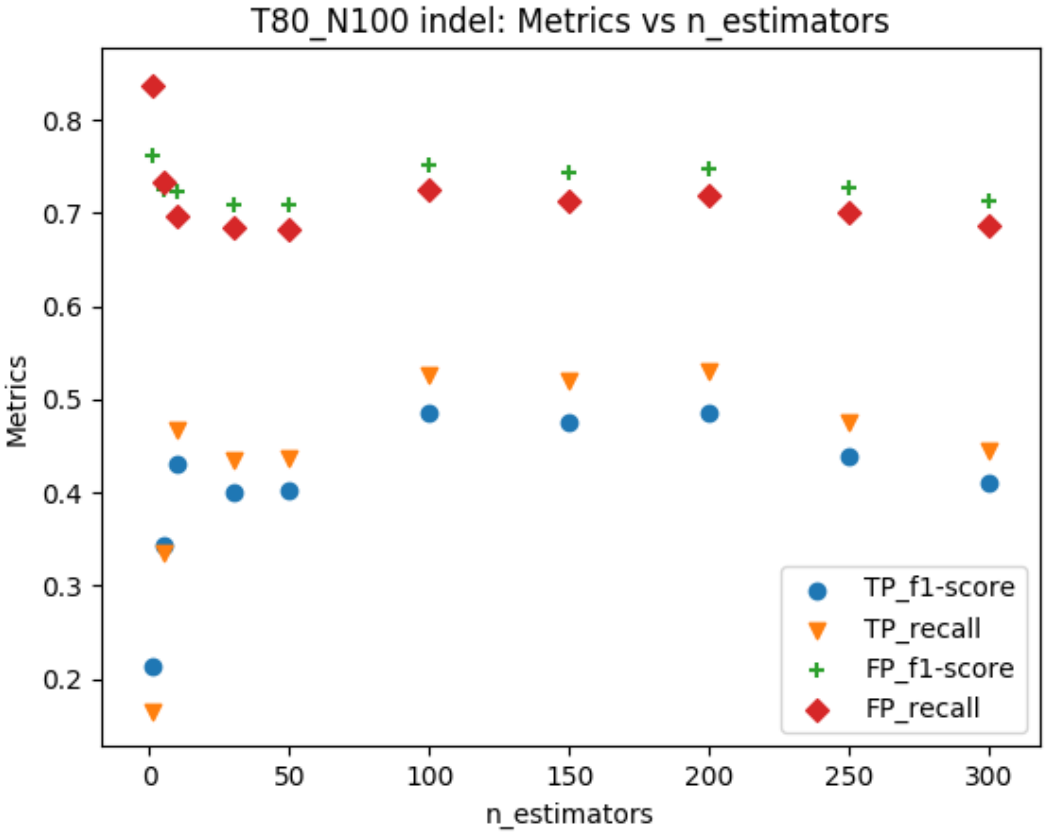


Figura 4.7: Influência do número de árvores, considerando os *indels* presentes na base *T80_N100*, nas métricas de revocação da classe mutação (*TP_recall*, triângulo em laranja), revocação da classe não mutação (*FP_recall*, diamante em vermelho), *f1-score* da classe mutação (*TP_f1-score*, círculo em azul) e *f1-score* da classe não mutação (*FP_f1-score*, sinal de adição em verde).

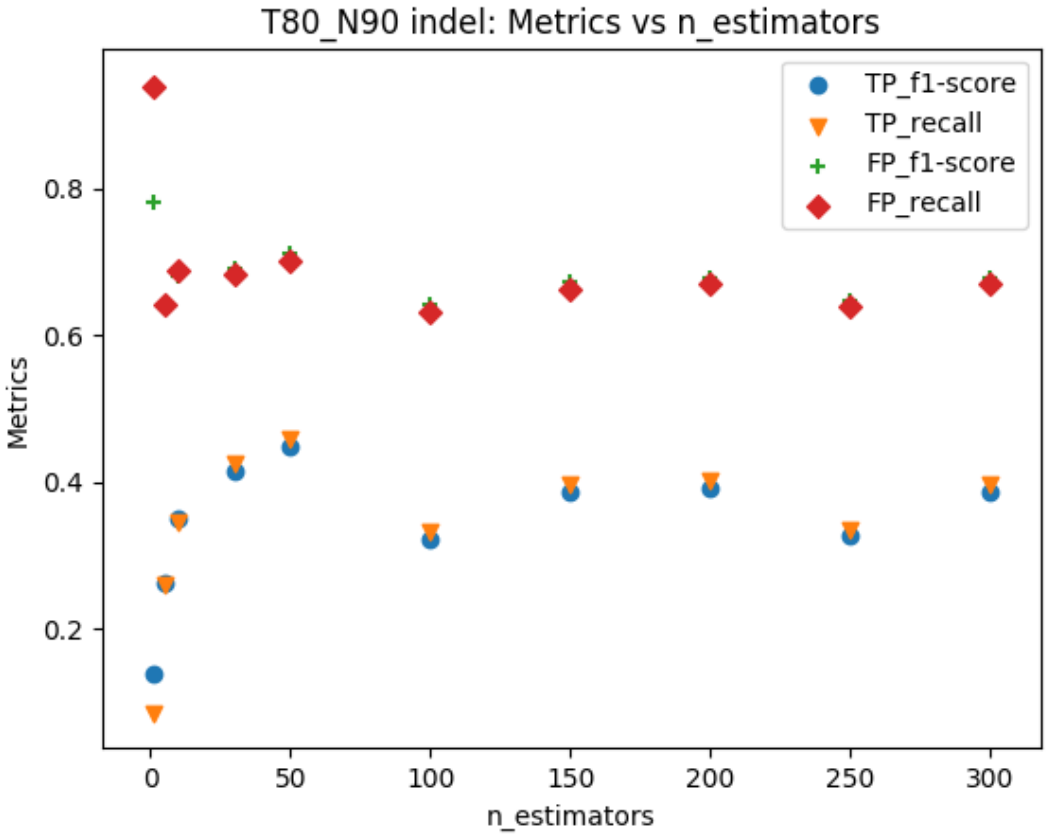


Figura 4.8: Influência do número de árvores, considerando os *indels* presentes na base *T80_N90*, nas métricas de revocação da classe mutação (*TP_recall*, triângulo em laranja), revocação da classe não mutação (*FP_recall*, diamante em vermelho), *f1-score* da classe mutação (*TP_f1-score*, círculo em azul) e *f1-score* da classe não mutação (*FP_f1-score*, sinal de adição em verde).

4.1.5 Testes adicionais

A partir dos testes definidos na Seção 3.5.5, realizou-se uma busca pelos melhores resultados utilizando entre 50 e 100 estimadores, fixando a contaminação em 0.35, total de atributos por árvore como 1 e 8 amostras por estimador. Após esse procedimento, o número de estimadores que apresentaram os melhores resultados para a classe de mutações (*TP*) foi fixado para realizar um novo teste de contaminação utilizando 3 valores distintos: 0.5, opção 'auto' (descrita na documentação do *Scikit-learn* (PEDREGOSA et al., 2011)) e a proporção de mutações descrita nas Tabelas 3.1 a 3.2. Os resultados obtidos são exibidos a seguir.

n_estimators entre 50 e 100

Nas Figuras 4.9 a 4.10 é possível perceber que 60 ADs resultam nas melhores métricas para classe *TP* considerando o conjunto *T80_N90* e *T80_N100*. Logo, para o segundo teste de contaminação, 60 ADs serão utilizadas, combinadas com 1 atributo por árvore ($max\_features = 0.1$), sendo ela treinada com um total de 8 amostras ($max\_samples = 8$).

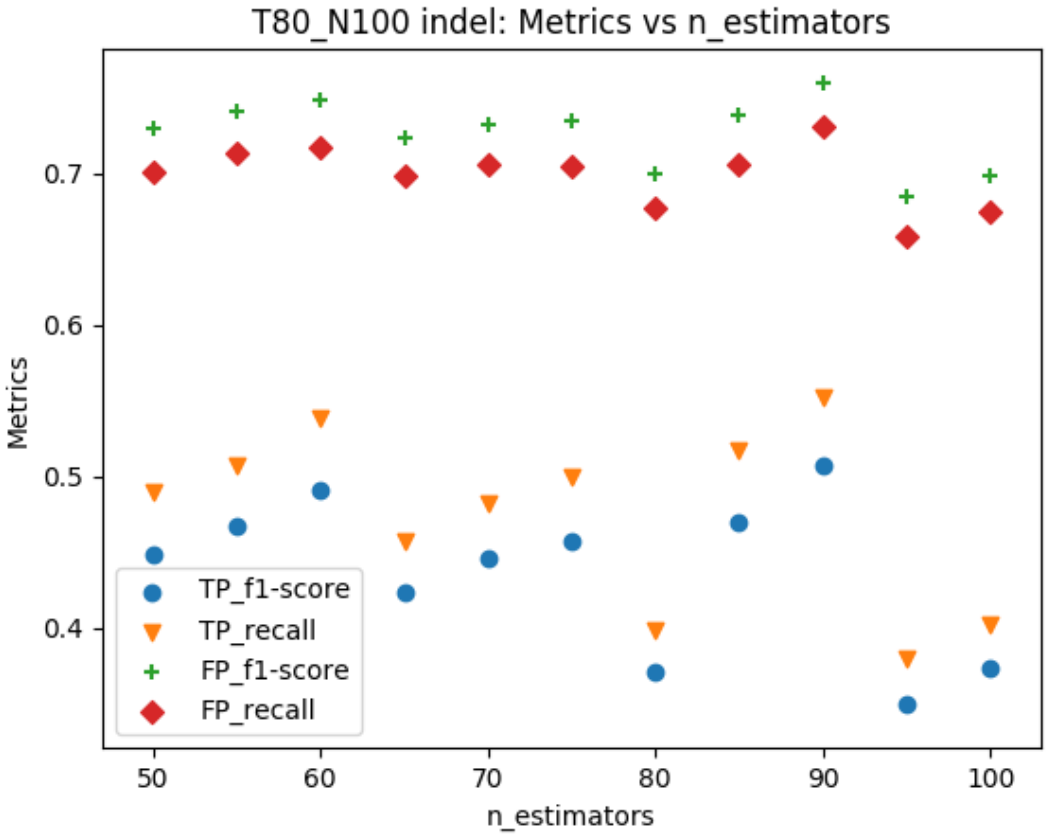


Figura 4.9: Influência do número de árvores entre 50 e 100, considerando os *indels* presentes na base *T80_N100*, nas métricas de revocação da classe mutação (*TP_recall*, triângulo em laranja), revocação da classe não mutação (*FP_recall*, diamante em vermelho), *f1-score* da classe mutação (*TP_f1-score*, círculo em azul) e *f1-score* da classe não mutação (*FP_f1-score*, sinal de adição em verde).

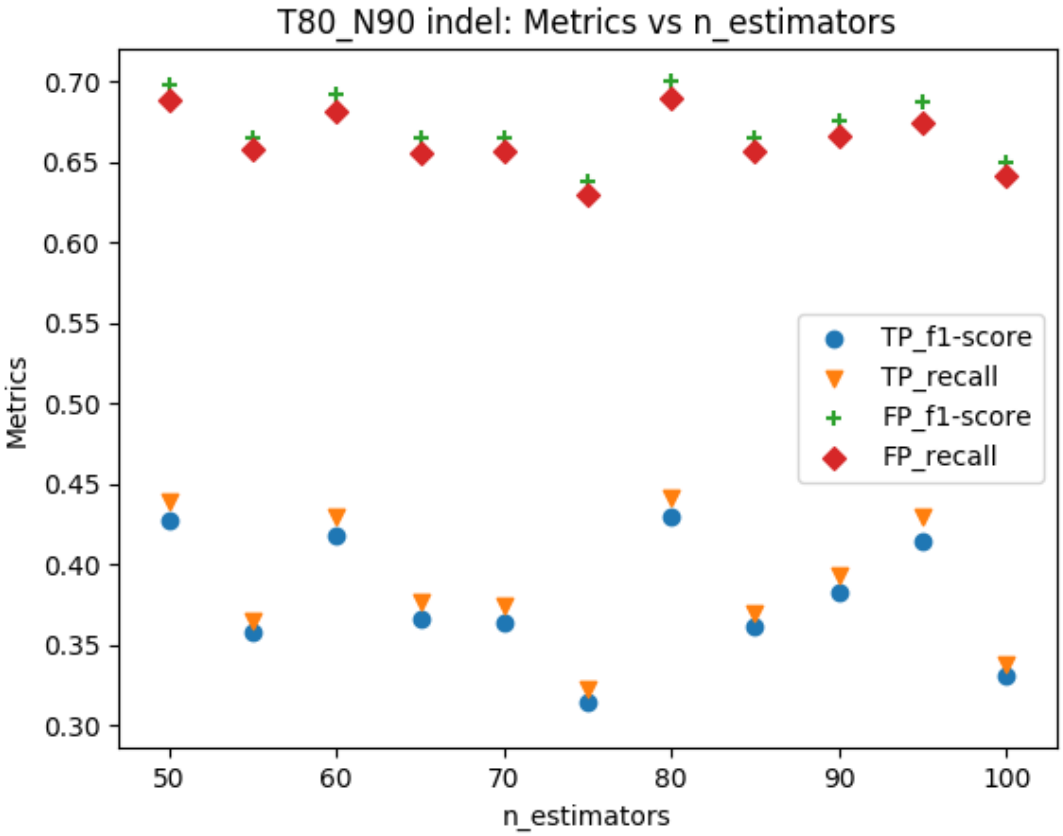


Figura 4.10: Influência do número de árvores entre 50 e 100, considerando os *indels* presentes na base *T80_N90*, nas métricas de revocação da classe mutação (*TP_recall*, triângulo em laranja), revocação da classe não mutação (*FP_recall*, diamante em vermelho), *f1-score* da classe mutação (*TP_f1-score*, círculo em azul) e *f1-score* da classe não mutação (*FP_f1-score*, sinal de adição em verde).

Segundo teste de contaminação

Considerando os resultados presentes nas Tabelas 4.1 a 4.2, observa-se que, em ambas as bases, a contaminação configurada para 'auto' gerou os melhores resultados em termos de revocação e *f1-score* para a classe de mutações, enquanto que a configuração que utilizava o percentual de mutações presentes nos dados apresentou os melhores resultados para a detecção de dados que não são mutações de fato.

Como o objetivo do trabalho é comparar os sistemas de detecção de mutações independentemente da base, escolheu-se contaminação igual a 0.35 para comparação com o modelo do *Random Forest*, uma vez que ela proporcionou um equilíbrio entre a qualidade da separação entre mutações e não mutações.

A Tabela 4.3 resume a configuração final dos parâmetros considerando a detecção de *indels*, bem como as variações percentuais (ganhos ou perdas) com relação às configura-

Tabela 4.1: Revocação e *f1-score* para mutações e não mutações, considerando a base de teste de *indels T80_N100* com $\frac{2}{3}$ do total de dados de teste considerando os pontos adicionais utilizados e os melhores parâmetros obtidos anteriormente (otimização).

contaminação	*f1-score* não mutação	revocação não mutação	*f1-score* mutação	revocação mutação
0.2908	0.7283	0.7338	0.3522	0.3460
0.5	0.6207	0.5303	0.4373	0.5853
auto	0.6420	0.5045	0.5635	0.8439
0.35 (otimização)	0.7487	0.7173	0.4903	0.5381

Tabela 4.2: Revocação e *f1-score* para mutações e não mutações, considerando a base de teste de *indels T80_N90* com $\frac{2}{3}$ do total de dados de teste considerando os pontos adicionais utilizados e os melhores parâmetros obtidos anteriormente (otimização).

contaminação	*f1-score* não mutação	revocação não mutação	*f1-score* mutação	revocação mutação
0.3233	0.6901	0.6948	0.3749	0.3700
0.5	0.5977	0.5214	0.4461	0.5588
auto	0.4751	0.3622	0.4618	0.6787
0.35 (otimização)	0.6918	0.6813	0.4174	0.4299

ções padrão do *Isolation Forest*. A partir dela é possível perceber que a busca realizada conseguiu proporcionar melhoras significativas para a separação da classe de mutações, causando uma pequena redução nos dados que não eram mutações.

Tabela 4.3: Comparação dos resultados obtidos, em termos de revocação e *f1-score*, para as melhores configurações obtidas com relação aos valores padrão estabelecidos para o *Isolation Forest* considerando mutações do tipo *indels*. As variações percentuais (ganho ou perda com relação aos valores padrão) estão entre parênteses, sendo que 'x' significa não se aplica.

Base	Configuração	*f1-score* não mutação	revocação não mutação	*f1-score* mutação	revocação mutação
T80_N100	Padrão	0.7505 (x)	0.8400 (x)	0.1024 (x)	0.0740 (x)
T80_N100	Otimização	0.7487 (-0,2372%)	0.7173 (-14,6043%)	0.4903 (378,9566%)	0.5381 (627,1486%)
T80_N90	Padrão	0.7304 (x)	0.8320 (x)	0.1627 (x)	0.1180 (x)
T80_N90	Otimização	0.6918 (-5,2846%)	0.6813 (-18,1106%)	0.4174 (156,6370%)	0.4299 (264,5722%)

4.2 SNV

O segundo tipo de mutação a ser investigado foram SNVs. A ordem dos parâmetros, bem como os valores nos testes iniciais, segue de acordo com o definido na Tabela 3.4.

4.2.1 Variável: contaminação (*contamination*)

De maneira análoga aos resultados obtidos para *indels*, a contaminação com relação aos SNVs se mostrou diretamente proporcional com relação às amostras de mutação (TP) e inversamente proporcional à outra classe de não mutação (FP), bem como os valores das

métricas em cada contaminação analisada. O comportamento desse parâmetro é mostrado nas Figuras 4.11 a 4.12.

Uma contaminação de 0.35 foi adotada devido aos mesmos critérios adotados para *indels* (maior revocação de *TP* que mantém essa métrica maior o igual a 0.6 para *FP*) para prosseguir com a avaliação do próximo parâmetro (*max_features*).

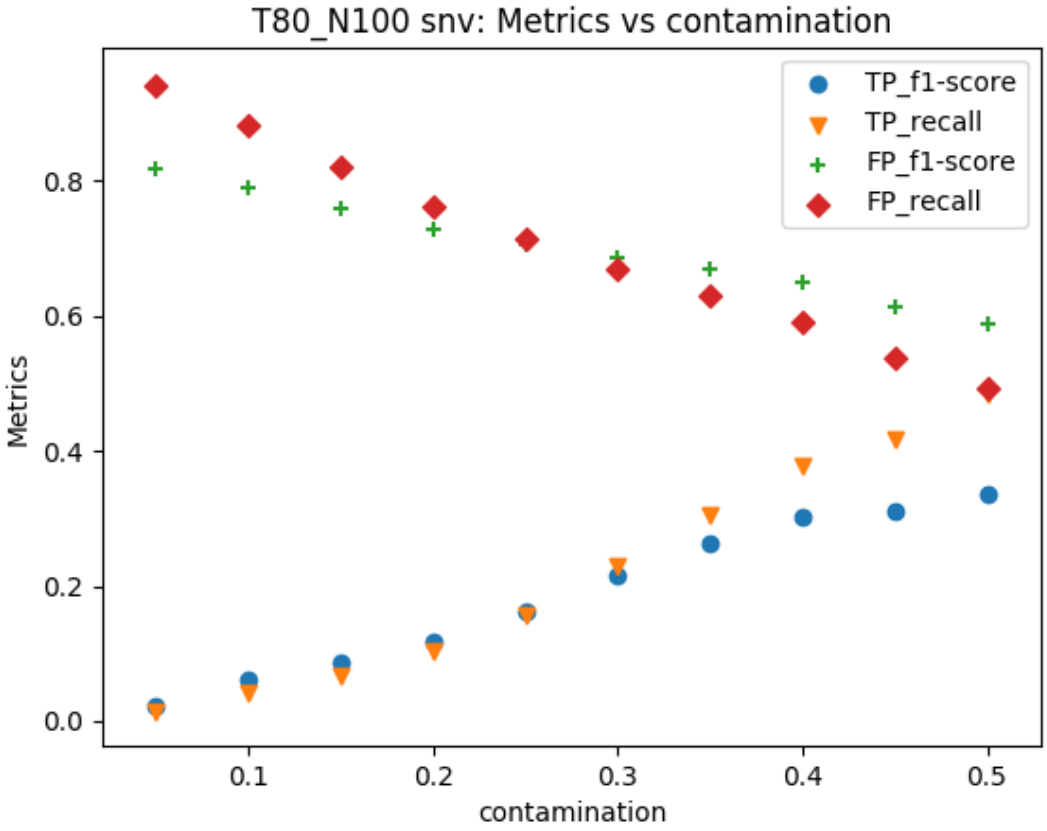


Figura 4.11: Influência da contaminação , considerando os SNVs presentes na base *T80_N100*, nas métricas de revocação da classe mutação (*TP_recall*, triângulo em laranja), revocação da classe não mutação (*FP_recall*, diamante em vermelho), *f1-score* da classe mutação (*TP_f1-score*, círculo em azul) e *f1-score* da classe não mutação (*FP_f1-score*, sinal de adição em verde).

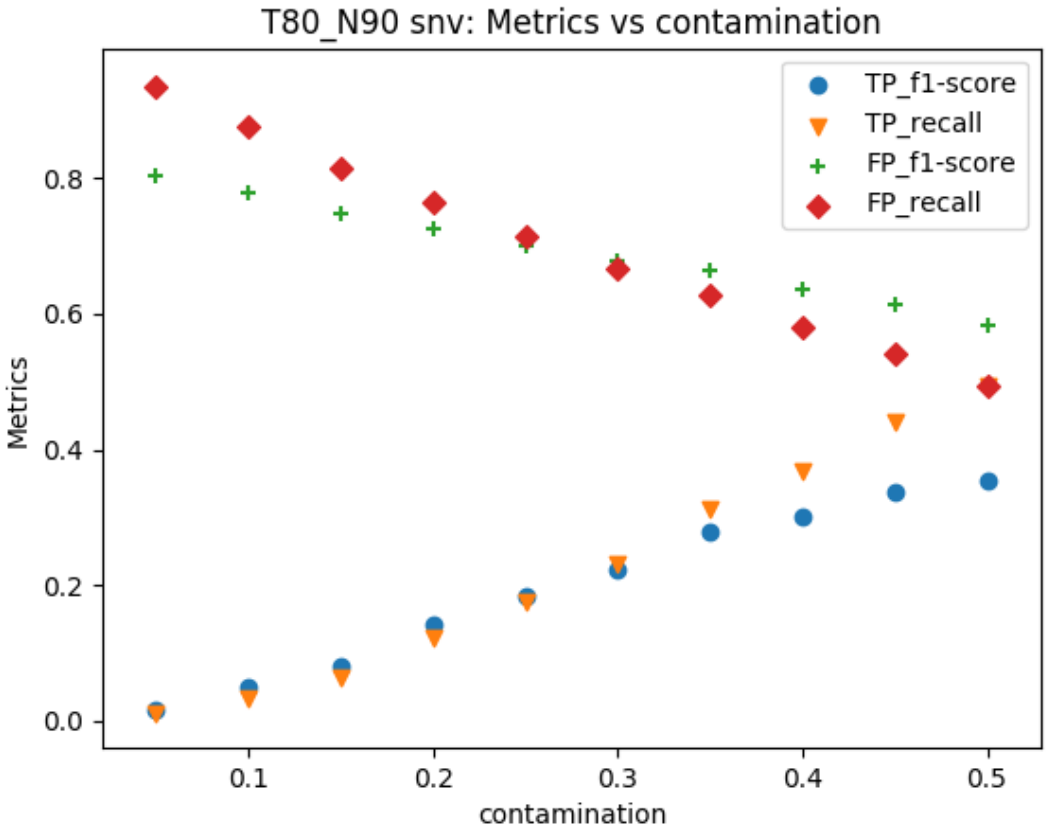


Figura 4.12: Influência da contaminação, considerando os SNVs presentes na base *T80_N90*, nas métricas de revocação da classe mutação (*TP_recall*, triângulo em laranja), revocação da classe não mutação (*FP_recall*, diamante em vermelho), *f1-score* da classe mutação (*TP_f1-score*, círculo em azul) e *f1-score* da classe não mutação (*FP_f1-score*, sinal de adição em verde).

4.2.2 Variável: número máximo de atributos (*max_features*)

O número máximo de atributos utilizados para detectar mutações do tipo SNV foi avaliado após fixar a contaminação em 0.35 e, diferentemente do apresentado para *indels* (Figuras 4.3 a 4.4), utilizar todos os atributos disponíveis (10 no total) resultou em um dos melhores parâmetros para a classe *TP* (mutações). Os 10 atributos para SNVs foram escolhidos, pois, de acordo com o exibido nas Figuras 4.13 a 4.14, pode-se perceber um comportamento crescente das métricas de ambas as classes com relação ao número de atributos utilizados. Assim, espera-se que utilizar o máximo de atributos disponíveis resulte em média nas melhores métricas.

É interessante notar que, para os SNVs, o número máximo de atributos utilizados por cada uma das ADs não causou impacto significativo na qualidade da detecção somática, uma vez que as métricas para a classe *TP* continuaram abaixo de 0.3, enquanto que o

mesmo teste para *indels* fez com que os resultados aumentassem em aproximadamente 10% para a classe *TP*.

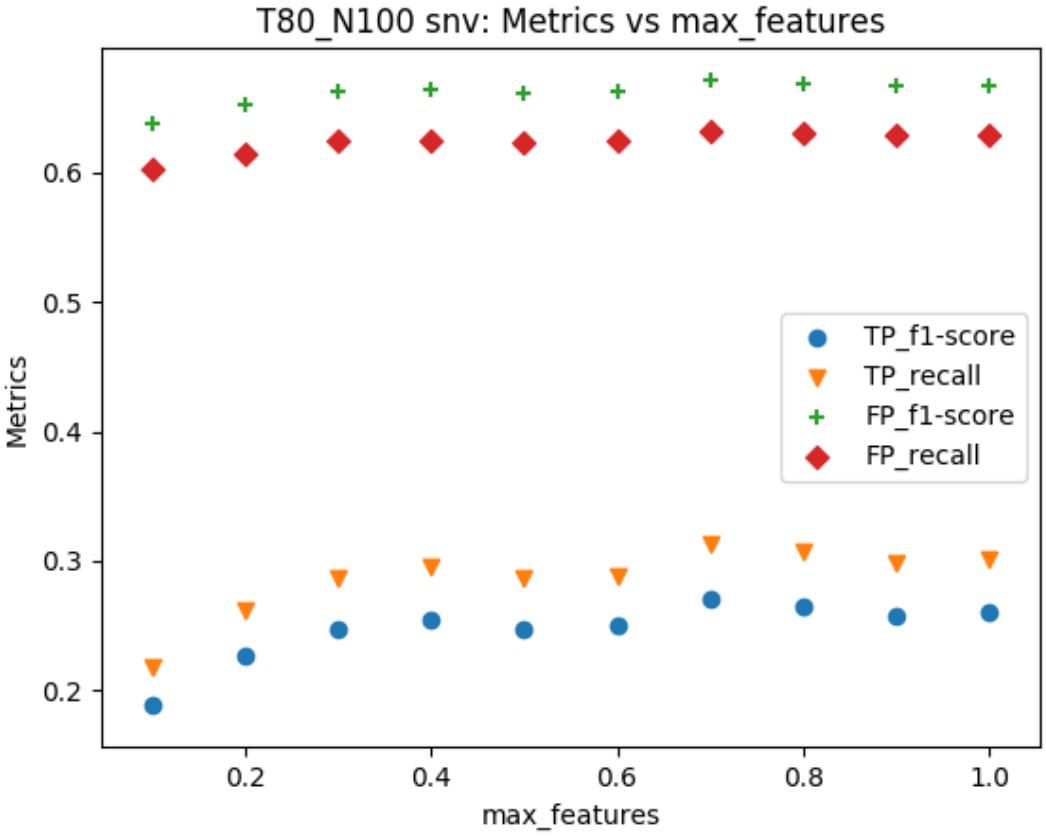


Figura 4.13: Influência do número máximo de atributos utilizados para treinar cada árvore, considerando os SNVs presentes na base *T80_N100*, nas métricas de revocação da classe mutação (*TP_recall*, triângulo em laranja), revocação da classe não mutação (*FP_recall*, diamante em vermelho), *f1-score* da classe mutação (*TP_f1-score*, círculo em azul) e *f1-score* da classe não mutação (*FP_f1-score*, sinal de adição em verde).

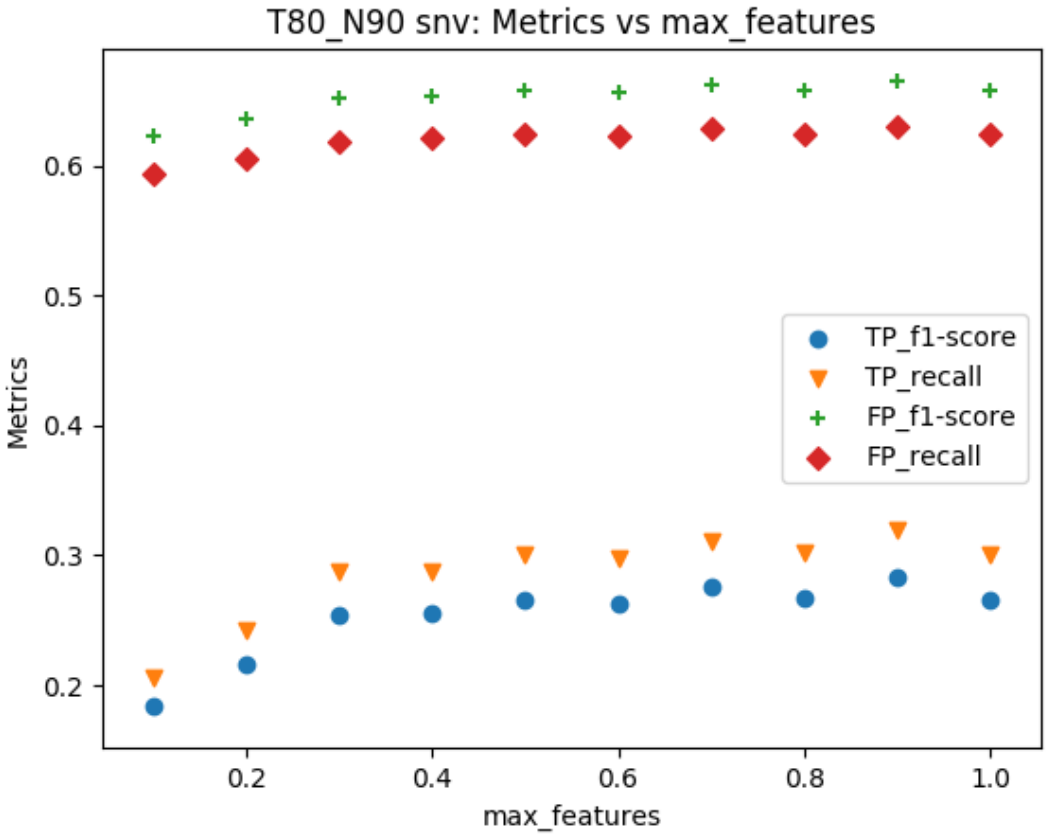


Figura 4.14: Influência do número máximo de atributos utilizados para treinar cada árvore, considerando os SNVs presentes na base *T80_N90*, nas métricas de revocação da classe mutação (*TP_recall*, triângulo em laranja), revocação da classe não mutação (*FP_recall*, diamante em vermelho), *f1-score* da classe mutação (*TP_f1-score*, círculo em azul) e *f1-score* da classe não mutação (*FP_f1-score*, sinal de adição em verde).

4.2.3 Variável: número máximo de amostras (*max_samples*)

Utilizando uma contaminação de 0.35 e todos os atributos disponíveis para a tarefa de detectar SNVs, o total de amostras empregadas para criar cada uma das ADs foi avaliado. Como mostram as Figuras 4.15 a 4.16, as métricas para a classe *TP* na base *T80_N100* apresentam um comportamento inversamente proporcional aos valores de *max_samples*, enquanto que os dados presentes em *T80_N90* evidenciam comportamento diretamente proporcional com relação a essas variáveis. Essa característica torna difícil a escolha de um valor comum para ambas as bases, exigindo que perdas de desempenho na revocação ou em *f1-score* sejam aceitas. Na tentativa de minimizar esse efeito, $max\_samples = 128$ foi escolhido, uma vez que esse foi o primeiro parâmetro em comum entre as bases que apresentou resultados satisfatórios.

De maneira semelhante à análise de *indels*, a escala mono-log (eixo dos parâmetros

em escala logarítmica e eixo das métricas em escala linear) foi empregada para facilitar a compreensão de dados espaçados exponencialmente.

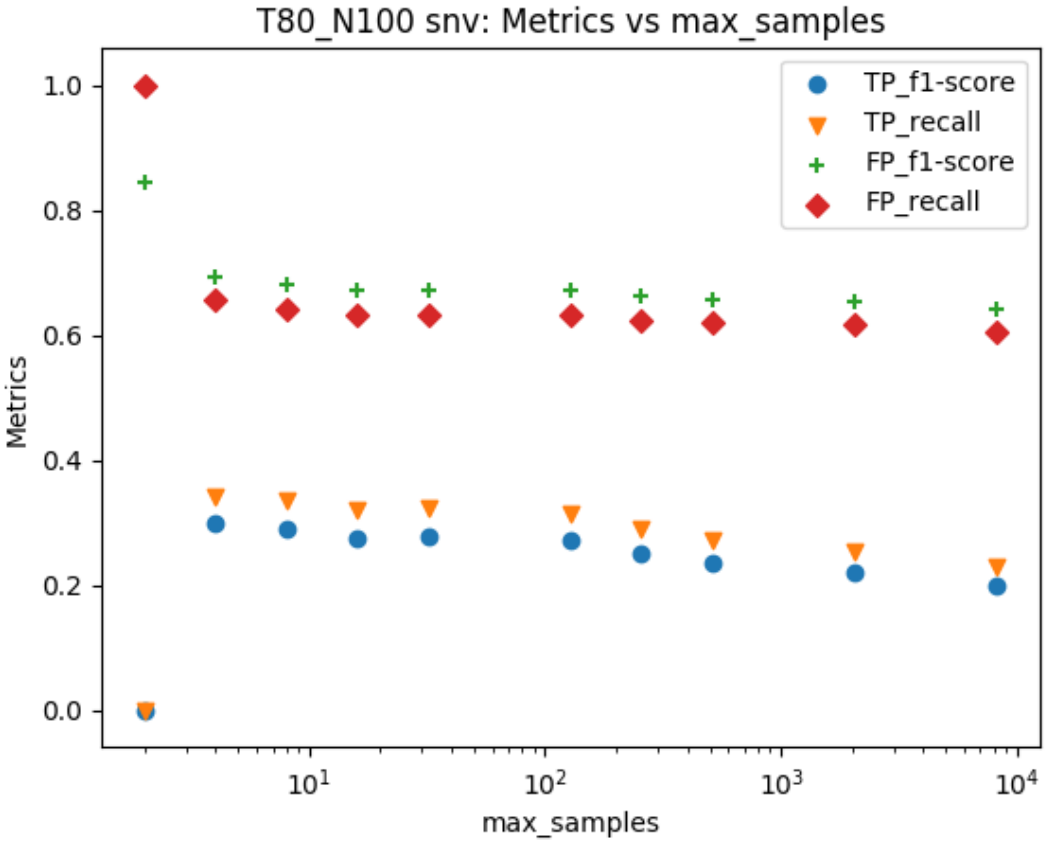


Figura 4.15: Influência do tamanho do conjunto de amostragem utilizado para treinar cada estimador, considerando os SNVs presentes na base *T80_N100*, nas métricas de revocação da classe mutação (*TP_recall*, triângulo em laranja), revocação da classe não mutação (*FP_recall*, diamante em vermelho), *f1-score* da classe mutação (*TP_f1-score*, círculo em azul) e *f1-score* da classe não mutação (*FP_f1-score*, sinal de adição em verde).

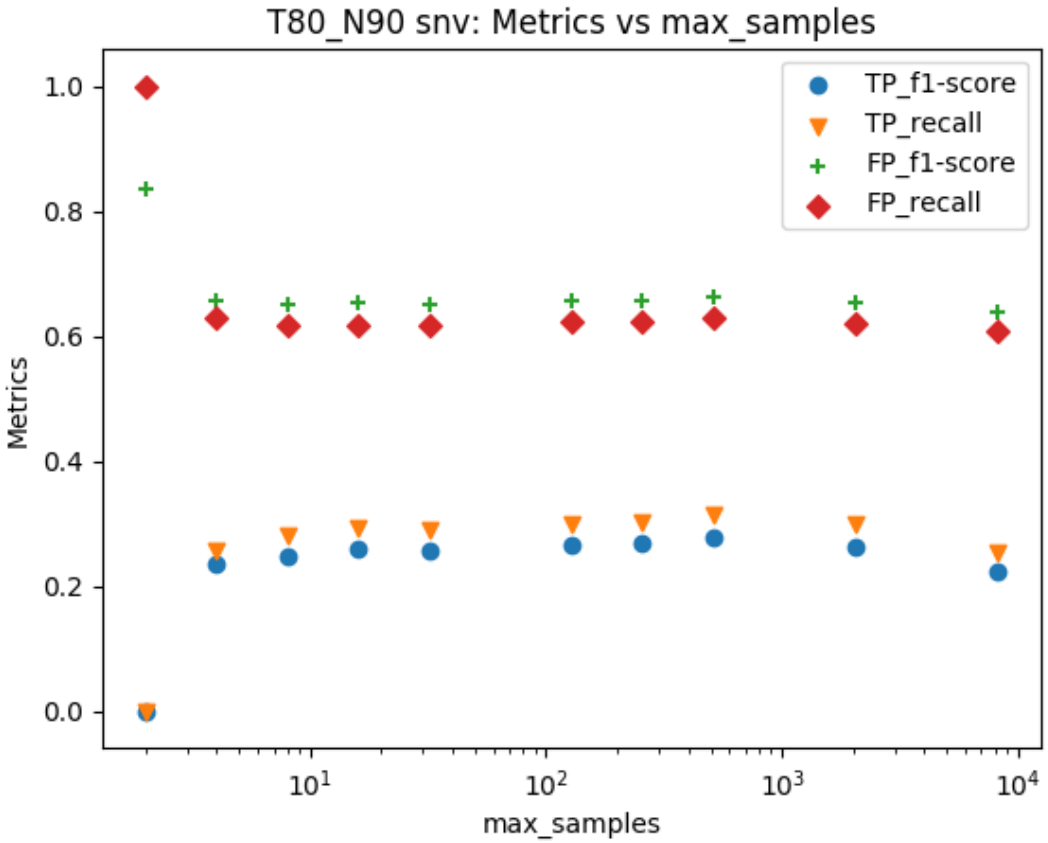


Figura 4.16: Influência do tamanho do conjunto de amostragem utilizado para treinar cada estimador, considerando os SNVs presentes na base *T80_N90*, nas métricas de revocação da classe mutação (*TP_recall*, triângulo em laranja), revocação da classe não mutação (*FP_recall*, diamante em vermelho), *f1-score* da classe mutação (*TP_f1-score*, círculo em azul) e *f1-score* da classe não mutação (*FP_f1-score*, sinal de adição em verde).

4.2.4 Variável: total de árvores utilizadas (*n_estimators*)

Considerando a análise inicial de parâmetros, a quantidade de árvores utilizada na tarefa de detecção de mutações somáticas foi avaliada. De forma similar ao apresentado para *indels* (Figuras 4.7 a 4.8), os melhores desempenhos com relação às métricas para a classe *TP* foram obtidos entre 50 e 100 estimadores como mostram as Figuras 4.17 a 4.18. Diferentemente do ocorrido para os *indels*, o melhor número de ADs para SNVs não foi tão destacado com relação aos demais valores testados.

Visto que essa variável impacta o tempo necessário para treinar o modelo e classificar dados novos, uma análise mais detalhada entre 50 e 100 estimadores foi realizada para verificar se havia a possibilidade de melhorar a classificação do *Isolation Forest* para SNVs mantendo sem aumentar significativamente a complexidade do modelo.

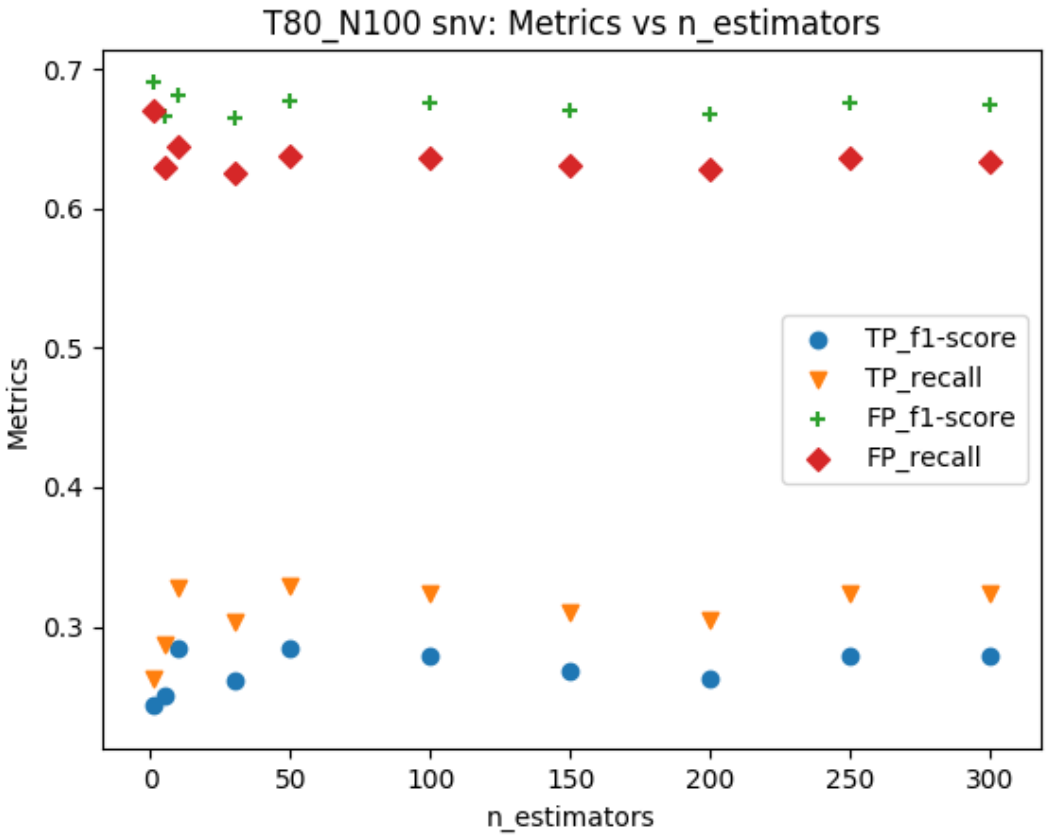


Figura 4.17: Influência do número de estimadores, considerando os SNVs presentes na base *T80_N100*, nas métricas de revocação da classe mutação (*TP_recall*, triângulo em laranja), revocação da classe não mutação (*FP_recall*, diamante em vermelho), *f1-score* da classe mutação (*TP_f1-score*, círculo em azul) e *f1-score* da classe não mutação (*FP_f1-score*, sinal de adição em verde).

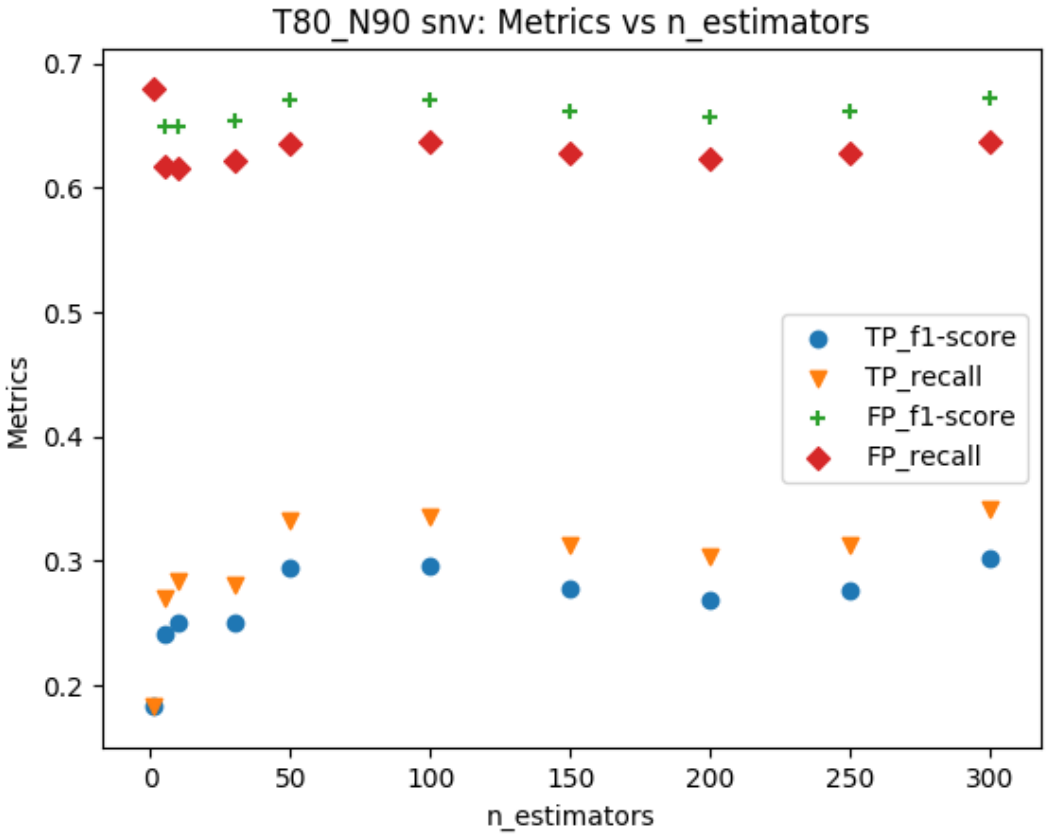


Figura 4.18: Influência do número de estimadores, considerando os SNVs presentes na base *T80_N90*, nas métricas de revocação da classe mutação (*TP_recall*, triângulo em laranja), revocação da classe não mutação (*FP_recall*, diamante em vermelho), *f1-score* da classe mutação (*TP_f1-score*, círculo em azul) e *f1-score* da classe não mutação (*FP_f1-score*, sinal de adição em verde).

4.2.5 Testes adicionais

A partir dos testes definidos na Seção 3.5.5, realizou-se uma busca pelos melhores resultados utilizando entre 50 e 100 estimadores, fixando a contaminação em 0.35, total de atributos por árvore como 1.0 (todos os 10 atributos disponíveis para SNVs) e 128 amostras por estimador. Após esse procedimento, o número de estimadores que apresentaram os melhores resultados para a classe de mutações (*TP*) foi fixado para realizar um novo teste de contaminação utilizando 3 valores distintos: 0.5, opção 'auto' (descrita na documentação do *Scikit-learn* (PEDREGOSA et al., 2011)) e a proporção de mutações descrita nas Tabelas 3.1 a 3.2. Os resultados obtidos são exibidos a seguir.

n_estimators entre 50 e 100

O experimento realizado com o número de estimadores entre 50 e 100 (11 valores, iniciando em 50 e incluindo 100 com passo de 5) para SNVs não apresentou resultados melhores dentro deste intervalo se comparado ao obtido nas Figuras 4.17 a 4.18. Por isso, 50 estimadores foram utilizados como sendo os melhores parâmetros para a detecção de SNVs.

Segundo teste de contaminação

Considerando os resultados presentes nas Tabelas 4.4 a 4.5, observa-se que a contaminação 0.5 gerou os melhores resultados em termos de revocação e *f1-score* para a classe de mutações, enquanto que a configuração 'auto' apresentou os melhores resultados para a detecção de dados que não são mutações de fato. Como o objetivo do trabalho é comparar os sistemas de detecção de mutações e se verificou que nenhum resultado apresentou melhorias na separação de ambas as classes simultaneamente, escolheu-se o ponto inicial de busca ($contaminação = 0.35$) para comparação com o modelo do *Random Forest*.

Tabela 4.4: Revocação e *f1-score* para mutações e não mutações, considerando a base de teste de SNVs *T80_N100* com $\frac{2}{3}$ do total de dados de teste considerando os pontos adicionais utilizados e os melhores parâmetros obtidos anteriormente (otimização).

contaminação	*f1-score* não mutação	revocação não mutação	*f1-score* mutação	revocação mutação
0.5	0.5823	0.4894	0.3315	0.4761
auto	0.7612	0.8266	0.0732	0.0560
0.2609	0.7029	0.7044	0.1806	0.1795
0.35 (otimização)	0.6779	0.6385	0.2839	0.3290

Tabela 4.5: Revocação e *f1-score* para mutações e não mutações, considerando a base de teste de SNVs *T80_N90* com $\frac{2}{3}$ do total de dados de teste considerando os pontos adicionais utilizados e os melhores parâmetros obtidos anteriormente (otimização).

contaminação	*f1-score* não mutação	revocação não mutação	*f1-score* mutação	revocação mutação
0.5	0.6203	0.5243	0.4139	0.5771
auto	0.7580	0.8383	0.0665	0.0486
0.2745	0.6915	0.6926	0.2095	0.2085
0.35 (otimização)	0.6699	0.6355	0.2946	0.3331

A Tabela 4.6 resume a configuração final dos parâmetros considerando a detecção de SNVs, bem como as variações percentuais (ganhos ou perdas) com relação às configurações padrão do *Isolation Forest*. A partir dela é possível perceber que a configuração padrão acarreta perda da qualidade de detecções das mutações, obtendo-se bons resultados somente para os dados que não eram relacionados a mutações de fato. Além disso, os ganhos obtidos para a classe de mutações utilizando parâmetros otimizados foram muito superiores às perdas acarretadas na classe de não mutações.

Tabela 4.6: Comparação dos resultados obtidos, em termos de revocação e *f1-score*, para as melhores configurações obtidas com relação aos valores padrão estabelecidos para o *Isolation Forest* considerando mutações do tipo SNVs. As variações percentuais (ganho ou perda com relação aos valores padrão) estão entre parênteses, sendo que 'x' significa não se aplica.

Base	Configuração	*f1-score* não mutação	revocação não mutação	*f1-score* mutação	revocação mutação
T80_N100	Padrão	0.7827 (x)	0.8673 (x)	0.0649 (x)	0.0457 (x)
T80_N100	Otimização	0.6779 (-13.3853%)	0.6385 (-26.3727%)	0.2839 (337.4730%)	0.3290 (619.7331%)
T80_N90	Padrão	0.7622 (x)	0.8478 (x)	0.0566 (x)	0.0404 (x)
T80_N90	Otimização	0.6699 (-12.1074%)	0.6355 (-25.0360%)	0.2946 (420.7317%)	0.3331 (724.5297%)

4.3 Comparação *Isolation Forest* e *Random Forest*

As Tabelas 4.7 a 4.8 contêm, respectivamente, os resultados obtidos para *indels* e SNVs, em termos de revocação e *f1-score*, ao utilizar os modelos de AM *Isolation Forest* e *Random Forest*. Observando os resultados, pode-se perceber que o *Random Forest* consegue separar muito melhor dados que representam mutações de fato dos que não representam, visto que as revocações e *f1-scores* obtidas para todas as classes em ambas as bases superam 90% de acerto.

Considerando as mesmas métricas para o *Isolation Forest*, nota-se que a melhor métrica para *indels* foi obtida pela métrica *f1-score* para os dados que não eram mutações na base *T8_N100* (0.74872), enquanto que SNVs também apresentaram melhor resultado nessa medida de avaliação (0.67789).

Tabela 4.7: Revocações e *f1-score* para as classes de mutações e não mutações obtidas para *indels* ao utilizar os modelos de *Isolation Forest* e *Random Forest* nas porções contendo $\frac{2}{3}$ dos dados de teste de *T80_N100* e *T80_N90*

Base	Classificador	*f1-score* não mutação	revocação não mutação	*f1-score* mutação	revocação mutação
T80_N100	Random Forest	0.9574	0.9411	0.9063	0.9420
T80_N100	Isolation Forest	0.7487	0.7173	0.4903	0.5381
T80_N90	Random Forest	0.9476	0.9221	0.9039	0.9502
T80_N90	Isolation Forest	0.6918	0.6813	0.4174	0.4299

Tabela 4.8: Revocações e *f1-score* para as classes de mutações e não mutações obtidas para SNVs ao utilizar os modelos de *Isolation Forest* e *Random Forest* nas porções contendo $\frac{2}{3}$ dos dados de teste de *T80_N100* e *T80_N90*

Base	Classificador	*f1-score* não mutação	revocação não mutação	*f1-score* mutação	revocação mutação
T80_N100	Random Forest	0.9911	0.9903	0.9757	0.9778
T80_N100	Isolation Forest	0.6779	0.6385	0.2839	0.3290
T80_N90	Random Forest	0.9903	0.9904	0.9751	0.9747
T80_N90	Isolation Forest	0.6699	0.6355	0.2946	0.3331

4.4 Discussão

Ao se observar os resultados obtidos, pode-se perceber que *indels* e SNVs tiveram comportamentos distintos com relação a cada um dos parâmetros avaliados (contaminação, quantidade de atributos e amostras por ADs, além da quantidade total de ADs utilizada).

Como indicam os resultados obtidos para *indels* (Figuras 4.1 a 4.10), as métricas com relação à classe de mutação começaram a obter níveis significativamente maiores a partir do número máximo de atributos por árvore em relação a aqueles proporcionados pela contaminação de 0.35. Se compararmos esse comportamento levando em consideração SNVs, percebe-se que esses últimos (Figuras 4.11 a 4.18) se beneficiaram menos dos testes básicos dos parâmetros, uma vez que as métricas para não mutações se situaram entre 0.63 e 0.68 e para mutações, no intervalo entre 0.28 e 0.34 (Tabelas 4.4 a 4.5), enquanto que as mesmas métricas para *indels* ficaram, respectivamente, entre os intervalos 0.68-0.75 e 0.41-0.54 (Tabelas 4.1 a 4.2).

Um ponto interessante a se observar é o fato de que ambos *indels* e SNVs apresentaram comportamentos semelhantes com relação à contaminação (Figuras 4.1 a 4.2 e Figuras 4.11 a 4.12 respectivamente) e exibiram melhoras significativas com relação aos valores padrão dos parâmetros testados se considerada a classe de mutações (Tabela 4.3 e Tabela 4.6 respectivamente).

Comparando os resultados obtidos com o *Isolation Forest* com os do *Random Forest* (Tabelas 4.7 a 4.8), pode-se notar que o primeiro modelo apresentou resultados significativamente inferiores ao segundo (reduções maiores que 15% nas métricas de desempenho de classificação). Uma das hipóteses para explicar a diferença obtida pode ser o fato de que o *Random Forest* (JAMES et al., 2013) busca, a partir das classes fornecidas, encontrar os valores dos atributos que melhor separam os dados, enquanto que o *Isolation Forest* (LIU; TING; ZHOU, 2008) utiliza valores aleatórios para realizar essa divisão.

Embora as bases utilizadas sejam amplas e bem citadas, em bases e algoritmos futuros um dos modelos pode ser mais ou menos favorecidos dependendo das amostras. Esse caso deverá ser investigado no futuro.

Uma vez que a redução da qualidade da detecção de mutações foi considerável, é preferível utilizar o modelo de aprendizagem supervisionada proporcionado pelo *Random Forest* em detrimento da abordagem não supervisionada adotada pelo *Isolation Forest*.

Capítulo 5

Conclusão

A tarefa de detectar mutações somáticas a partir de alinhamentos de sequências de DNA se torna mais importante a cada dia dado à aplicabilidade na área médica e à crescente disponibilidade de grande quantidade de dados provenientes de máquinas de sequenciamento de nova geração. O enorme tamanho das sequências de DNA torna a análise manual extremamente custosa e demorada e, por isso, a pesquisa de métodos que consigam identificar mutações somáticas de forma automática pode ser de grande interesse para a área.

Considerando as abordagens computacionais desenvolvidas que buscam resolver esse problema, pode-se mencionar aquelas com base estocástica criadas por Kim et al. (2018), Fan et al. (2016) e Lai et al. (2016), sendo que o *Strelka2* adiciona um modelo de AM (*Random Forest*) à análise probabilística para aprimorar a qualidade da detecção de mutações somáticas em termos da precisão obtida. O emprego de abordagens puramente de AM, como feito em Sahraeian et al. (2019), ou de uma combinação, por meio de modelos de AM, de atributos de diferentes classificadores, incluindo o *Strelka2* (KIM et al., 2018), como descrito em Anzar et al. (2019) também são outras maneiras de detectar mutações somáticas.

Utilizando os atributos calculados pelo *Strelka2* (KIM et al., 2018), comparou-se o desempenho obtido, em termos das métricas de revocação e *f1-score*, pelos modelos de AM *Random Forest* (JAMES et al., 2013) e *Isolation Forest* (LIU; TING; ZHOU, 2008) abordagens de aprendizagem supervisionada e não supervisionada respectivamente, a fim de analisar por qual deles se obtém os melhores resultados.

Baseando-se nos resultados descritos no Capítulo 4, conclui-se que o *Random Forest* apresentou resultados significativamente melhores para identificar mutações após a filtragem estocástica realizada pelo *Strelka2* (KIM et al., 2018).

Dentre outras possibilidades, pode-se mencionar testar os parâmetros definidos no Capítulo 3 para a base *T20-N100* por uma questão de completude e realizar a otimização

dos parâmetros *max_features*, *max_samples* e *n_estimators* do *Isolation Forest* tomando com base contaminações de 0.5 e 'auto' a fim de investigar a possibilidade de aumentar a qualidade da detecção de mutações somáticas. Uma segunda abordagem a ser considerada para o *Isolation Forest* seria realizar uma contagem das bases que estão dentro de uma janela que inclui uma possível mutação somática, semelhante às matrizes de entrada do *Neusomatic* (SAHRAEIAN et al., 2019), e utilizar essas contagens como atributos para o *Isolation Forest*. Visto que modelos não supervisionados têm a vantagem de não precisarem de dados classificados para treinamento, é interessante avaliar se com o uso de outras formas desse tipo de aprendizagem como, por exemplo, mapas auto-organizáveis pode-se obter resultados semelhantes a modelos supervisionados na tarefa de detectar mutações somáticas.

Referências

AMABIS, J. M.; MARTHO, G. R. *Fundamentos da Biologia Moderna*. [S.l.]: Editora Moderna, 2006. v. 4.

ANZAR, I.; SVERCHKOVA, A.; STRATFORD, R.; CLANCY, T. Neomutate: an ensemble machine learning framework for the prediction of somatic mutations in cancer. *BMC Medical Genomics*, v. 12, n. 1, p. 63, 2019.

BUITINCK, L.; LOUPPE, G.; BLONDEL, M.; PEDREGOSA, F.; MUELLER, A.; GRISEL, O.; NICULAE, V.; PRETTENHOFER, P.; GRAMFORT, A.; GROBLER, J.; LAYTON, R.; VANDERPLAS, J.; JOLY, A.; HOLT, B.; VAROQUAUX, G. API design for machine learning software: experiences from the scikit-learn project. In: *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*. [S.l.: s.n.], 2013. p. 108–122.

CHEN, X.; SCHULZ-TRIEGLAFF, O.; SHAW, R.; BARNES, B.; SCHLESINGER, F.; KäLLBERG, M.; COX, A. J.; KRUGLYAK, S.; SAUNDERS, C. T. Manta: rapid detection of structural variants and indels for germline and cancer sequencing applications. *Bioinformatics*, v. 32, n. 8, p. 1220–1222, 2015.

FAN, Y.; XI, L.; HUGHES, D. S. T.; ZHANG, J.; ZHANG, J.; FUTREAL, P. A.; WHEELER, D. A.; WANG, W. Muse: accounting for tumor heterogeneity using a sample-specific error model improves sensitivity and specificity in mutation calling from sequencing data. *Genome Biology*, v. 17, n. 1, p. 178, 2016.

GRIFFITHS, A. J. F.; MILLER, J. H.; SUZUKI, D. T.; LEWONTIN, R. C.; GELBART, W. M. *An Introduction to Genetic Analysis.* [S.l.]: New York: W. H. Freeman, 2000. v. 7.

HE, K.; ZHANG, X.; REN, S.; SUN, J. Deep residual learning for image recognition. In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. [S.l.: s.n.], 2016.

HO, T. K. The random subspace method for constructing decision forests. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 20, n. 8, p. 832–844, 1998.

JAMES, G.; WITTEN, D.; HASTIE, T.; TIBSHIRANI, R. *An Introduction to Statistical Learning with Applications in R*. [S.l.]: Springer, 2013.

JUN, L.; SHUNYI, Z.; YANQING, L.; ZAILONG, Z. Internet traffic classification using machine learning. In: *2007 Second International Conference on Communications and Networking in China*. [S.l.: s.n.], 2007. p. 239–243.

KIM, S.; SCHEFFLER, K.; HALPERN, A. L.; BEKRITSKY, M. A.; NOH, E.; KÄLLBERG, M.; CHEN, X.; KIM, Y.; BEYTER, D.; KRUSCHE, P.; SAUNDERS, C. T. Strelka2: fast and accurate calling of germline and somatic variants. *Nature Methods*, v. 15, n. 8, p. 591–594, 2018.

LAI, Z.; MARKOVETS, A.; AHDESMAKI, M.; CHAPMAN, B.; HOFMANN, O.; MCEWEN, R.; JOHNSON, J.; DOUGHERTY, B.; BARRETT, J. C.; DRY, J. R. VarDict: a novel and versatile variant caller for next-generation sequencing in cancer research. *Nucleic Acids Research*, v. 44, n. 11, p. e108–e108, 2016.

LECUN, Y.; BOSER, B. E.; DENKER, J. S.; HENDERSON, D.; HOWARD, R. E.; HUBBARD, W. E.; JACKEL, L. D. Handwritten digit recognition with a back-propagation network. In: TOURETZKY, D. S. (Ed.). *Advances in Neural Information Processing Systems 2*. [S.l.]: Morgan-Kaufmann, 1990. p. 396–404.

LIBBRECHT, M. W.; NOBLE, W. S. Machine learning applications in genetics and genomics. *Nature Reviews Genetics*, v. 16, n. 6, p. 321–332, 2015.

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

MITCHELL, T. M. *Machine Learning*. [S.l.]: McGraw-Hill Science/Engineering/Math, 1997.

MOORTHIE, S.; MATTOCKS, C. J.; WRIGHT, C. F. Review of massively parallel dna sequencing technologies. *The HUGO journal*, Springer Netherlands, v. 5, n. 1-4, p. 1–12, 2011.

PEDREGOSA, F.; VAROQUAUX, G.; GRAMFORT, A.; MICHEL, V.; THIRION, B.; GRISEL, O.; BLONDEL, M.; PRETTENHOFER, P.; WEISS, R.; DUBOURG, V.; VANDERPLAS, J.; PASSOS, A.; COURNAPEAU, D.; BRUCHER, M.; PERROT, M.; DUCHESNAY, E. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, v. 12, p. 2825–2830, 2011.

PIROOZNIA, M.; YANG, J. Y.; YANG, M. Q.; DENG, Y. A comparative study of different machine learning methods on microarray gene expression data. *BMC Genomics*, v. 9, n. 1, p. S13, 2008.

SAHRAEIAN, S. M. E.; LIU, R.; LAU, B.; PODESTA, K.; MOHIYUDDIN, M.; LAM, H. Y. K. Deep convolutional neural networks for accurate somatic mutation detection. *Nature Communications*, v. 10, n. 1, p. 1041, 2019.

SHALEV-SHWARTZ, S.; BEN-DAVID, S. *Understanding Machine Learning: From Theory to Algorithms*. [S.l.]: Cambridge University Press, 2014.

VONSATTEL, J. P. G.; DIFIGLIA, M. Huntington disease. *Journal of neuropathology and experimental neurology*, v. 57, n. 5, p. 369–84, 1998.

XU, C. A review of somatic single nucleotide variant calling algorithms for next-generation sequencing data. *Computational and structural biotechnology journal*, Research Network of Computational and Structural Biotechnology, v. 16, p. 15–24, 2018.

ZHANG, H. Overview of sequence data formats. In: MATHÉ, E.; DAVIS, S. (Ed.). *Statistical Genomics*. [S.l.]: Humana Press, New York, NY, 2016, (Methods in Molecular Biology, v. 1418). p. 3–17.

ZHANG, X.; ZHAO, J.; LECUN, Y. Character-level convolutional networks for text classification. In: *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*. Cambridge, MA, USA: MIT Press, 2015. (NIPS'15), p. 649–657.