Spring 2022

Modeling Sequencing Artifacts for Next Generation Sequencing

Yvonna Leong
San Jose State University

Modeling Sequencing Artifacts for Next Generation Sequencing

A Project

Presented to

The Faculty of the Department of Computer Science

San José State University

In Fulfillment

of the Requirements for the Degree

Master of Science

By

Yvonna Leung

May 2022

ABSTRACT

Modeling Sequencing Artifacts for Next Generation Sequencing

by Yvonna Leung

Advancements in Next Generation Sequencing (NGS) have enabled detection of genetic alterations at large scales with high throughputs. NGS offers advantages over the established sequencing method, Sanger sequencing, by processing large sections of the genome simultaneously at a lower cost with higher accuracy. However, recent research has shown that sequencing artifacts are introduced at various steps in the NGS workflow. These artifacts are the result of an accumulation of errors from multiple steps, such as library preparation and downstream processes, and can result in variants being identified that aren't actually present in the sequenced genome. Therefore, there is a need to accurately distinguish between true variants and sequencing artifacts. This project included the building of a bioinformatics pipeline to process Whole Exome Sequencing (WES) datasets from the Sequence Read Archive (SRA), as well as a high-scale machine learning models to identify errors introduced in the genome sequencing process. Results showed that the models had high classification accuracy, ranging from 98% to 100%, as well as high precision and recall scores around 99% when positively identifying artifacts. One feature, "Allele Frequency" or 'AF', was shown to have powerful predictive power, with it alone able to accurately classify 99% of the training data. Since 'AF' is an important parameter in variant calling software, a further investigation was conducted, which found that values of 'AF' around 0.22 could correctly differentiate most artifacts from non-artifacts. Finally, another investigation was conducted into the predictive power of other features, and identified several other features capable of differentiating artifacts.

ACKNOWLEDGEMENTS

First and foremost, I would like to thank my advisor, Dr. Wendy Lee, for her invaluable technical advice, continuous support, and patience during the development of this Master's project. I would also like to extend my gratitude to my committee members, Dr. Philip Heller and Dr. William Andreopoulos, for their guidance and feedback. Their expertise and experience in the field have been of tremendous benefit to studies at San José State University. I would also like to thank members of the Lee Lab for their advice and feedback.

Last but not least, I would like to thank my husband and parents for their endless support and encouragement during my studies. Without them, I would not have been able to achieve my academic goals.

Contents

List of Figures

List of Tables

1 Introduction

1.1 Impact of DNA Sequencing on Genomics and Health

In recent decades, DNA sequencing has enabled a remarkable leap forward in knowledge of the human genome and its medical applications [3]. Advances in DNA sequencing enable more accurate disease diagnosis, as well as personalized treatment strategies for patients. Previously unsolved medical mysteries arising from numerous genetic factors are now diagnosable due to Next Generation Sequencing (NGS) [4]. Not only has increased access to genomic data improved the characterization of genetic disease, it has also enabled gene-based therapies and prevention [3].

Initial studies of the genome were inefficient, expensive, and technically challenging. Sequencing was a temperamental practice, and it would take several years to sequence single genes [5]. To address these issues, the Human Genome Project, started in 2003, began a massive scientific undertaking to sequence the entire human genome. Not only did the completion of this project usher in a new era in medicine and diagnostics, it also led to major advances in high-throughput DNA sequencing technology. Today, automated Sanger sequencing is still in use, primarily in clinical labs where it is acceptable to have low throughput, higher per-sample costs, and shorter sequencing reads. However, the cost of automated Sanger sequencing is too expensive for large-scale sequencing projects. In contrast, today's sequencing methods offer high speed, high scale, and low cost.

1.2 Next Generation Sequencing

Next generation sequencing (NGS), also known as "massively parallel sequencing", is capable of processing millions of reads in parallel. In contrast, Sanger sequencing, the previous fastest method, produces only one forward and reverse read per sequence [6]. It is much more limited in terms of speed and scalability, and took over a decade to decipher the human genome. Although NGS has mostly super-

seded Sanger sequencing, it has not been translated into routine clinical practice [7]. Sanger sequencing is still commonly used to validate variants, and is the gold standard with its 99.99% accuracy rate [6].

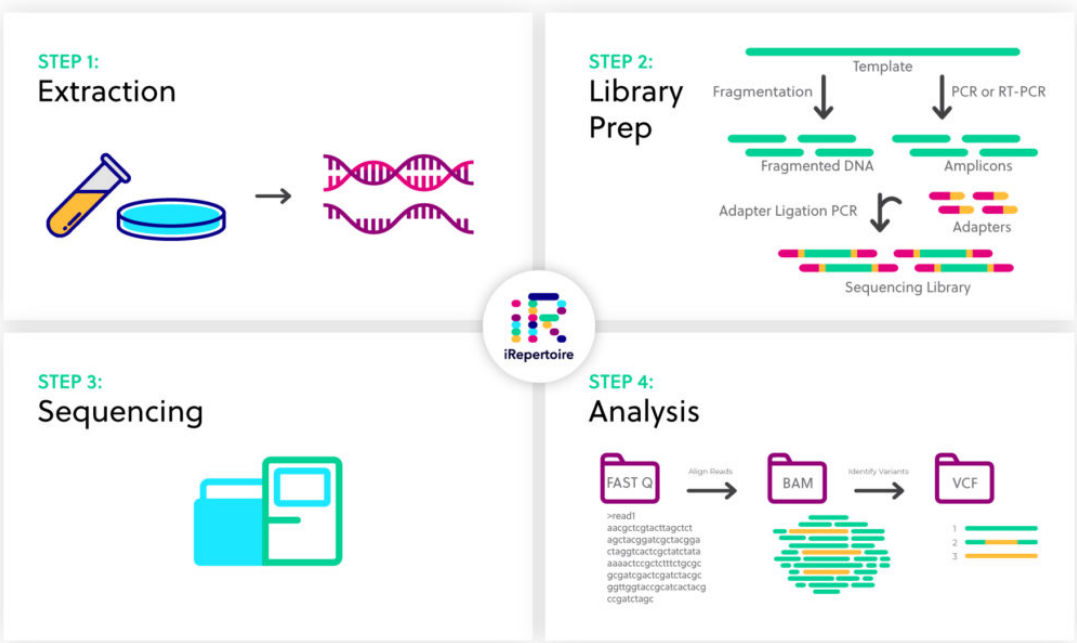

Figure 1: Next Generation Sequencing Workflow adapted from [1]

Figure 1 demonstrates that the NGS workflow contains four main steps: DNA extraction, library preparation, sequencing, and data analysis [8]. DNA extraction is the process of pulling genetic material from a sample. Extraction methods are generally designed to yield the largest possible quantity of nucleic acids from the sample. Purity of a sample is determined using spectrophotometers [1]. Next, the library preparation step makes DNA samples compatible with the sequencer through fragmentation and adapter ligation [8]. After loading DNA fragments to a flow cell, DNA fragments are clustered and amplified to generate millions of individual DNA strands [8]. These copies of DNA are then sequenced in parallel. After sequencing, the sequences are analyzed with various bioinformatics software tools to investigate sequence alignment, variant calling, data visualization, and much more.

1.3 Genomic Variants and Sequencing Artifacts

Genomic variants are areas of a genome that differ from a reference genome. A reference genome represents the genome of a single individual that has been cross-checked against the genomes of several other donors to ensure accuracy [9]. GRCh38 is the current reference genome build [9]. These include "single nucleotide polymorphisms" (SNPs), which are substitutions of single nucleotides, and "indels", which are insertions or deletions of one or more nucleotides. The process of identifying variants is called "variant calling". Most variants are SNPs, and they occur about once every 1000 nucleotides, meaning an average person has about four to five million SNPs in their genome [10]. Identifying variants can help predict how an individual will respond to drug treatment, or help determine the risk of developing certain diseases [10].

While variants occur naturally, sequencing artifacts are variations that are introduced through non-biological processes. Examples include SNPs and indels that are observed in the sequencing data but are not from the original biological samples. These occurrences are the consequence of accumulated errors from NGS workflow steps, such as library preparation kits and any downstream processes [11]. Sequence artifacts are often difficult to distinguish from true variants. This introduces a risk of false positive and false negative variant calls [12]. It is clinically important to identify whether a variant truly exists in the genome or is simply an artifact of the NGS process [12].

1.4 Causes of Artifacts

Sequencing artifacts from the NGS process can arise for a multitude of reasons. Understanding the sources of sequencing artifacts can improve the detection of true variants and help make better informed clinical decisions.

Because the NGS workflow has many steps, there are many opportunities where errors can occur that lead to sequencing artifacts. S. Haile noted that sequencing artifacts can also arise from DNA damage due to prior sample treatment and ex-

traction techniques with formalin and paraffin [12]. DNA fragmentation may also occur during longer storage time. These artifacts appear at a much higher rate compared with NGS libraries that were not prepared with formalin [12]. Sequencing artifacts can also be created through chemical modification or DNA damage through various molecular techniques. One source of artifacts arise from high levels of oxidation product, 8-oxoG, which are derived from an oxidative mechanism during high-powered DNA shearing [13]. PCR cycles are another major source of artifacts. PCR amplification bias can impact final sequence read counts, which may lead to an accumulation of artifacts in sequence reads [14]. Library sample preparation can also introduce artifacts. For example, N. Tanaka found that HyperPlus library kits generated significantly more SNV and indel artifacts compared with Agilent's SureSelect kit [11].

1.5 Genome in a Bottle

Having a well-characterized reference human genome allows for distinguishing sequencing artifacts from true biological variants. The National Institute of Standards and Technology (NIST) hosts a consortium known as the Genome in a Bottle (GIAB). This group develops reference materials and standards for benchmarking and characterizing human genome sequences [15]. The group has created high quality reference genomes, including the pilot genome NA12878, by utilizing multiple sequence datasets to correct for any systematic bias and to fill in sequence gaps using pooled data [16] [17]. Various data sets, sequencing technologies, and variant callers [16] were used to identify any high-confidence variant calls, which are variants that has been tested rigorously and have been identified as true variants [15]. NA12878 is the best-characterized and most-used reference sample for validating other human genome sequences, and was used as a reference genome in this study.

1.6 Sequence Read Archive

The Sequence Read Archive (SRA) is a database for housing publicly available NGS data, hosted by the National Center for Biotechnology Information (NCBI). For example, NA12878 can be found in SRA. Since its release in 2009, SRA has accumulated 9 million records, with a broad variety of attributes [18]. Metadata filters can also be used to find specific datasets for a researcher's needs, such as sample type, library layout, platform used, and experimental parameters [19].

1.7 Whole Genome Sequencing and Whole Exome Sequencing

Applications of NGS methods have skyrocketed with the advancement of targeted sequencing techniques, such as Whole Genome Sequencing (WGS) and Whole Exome Sequencing (WES). WGS determines the entire genome's sequence whereas WES only examines the exome regions of the genome, the coding regions that are transcribed and translated into proteins. Exomes only compose of 2% of the entire genome [20]. Although there are advantages to both approaches, the differences between them are large enough to impact diagnostic decisions. These differences remain a source of debate when determining which technique to use.

WGS provides uniform genome-wide coverage, but tends to be expensive and has a low depth of read coverage. The depth of read coverage refers to the number of sequencing reads aligned at a particular locus against the reference genome [21]. A low depth of read coverage provides lower confidence in calling variants. In contrast, WES is more cost-effective and provides higher depth of read coverage [21]. The higher read depth allows for higher confidence [20]. Figure 2 highlights the WES process of extracting exons from double-stranded DNA, also known as target enrichment [22]. Once target enrichment is completed, sequencing of the processed DNA occurs [22]. ClinVar is a publicly available database that reports information on germline and somatic variants and their genomic location [23]. Barbitoff, et al. found that more than 80% of variants reported in ClinVar were

from the exons, protein coding regions of the genome [24] and that many genetic mutations originate from these regions. Thus, WES was chosen for characterizing variants in this project.

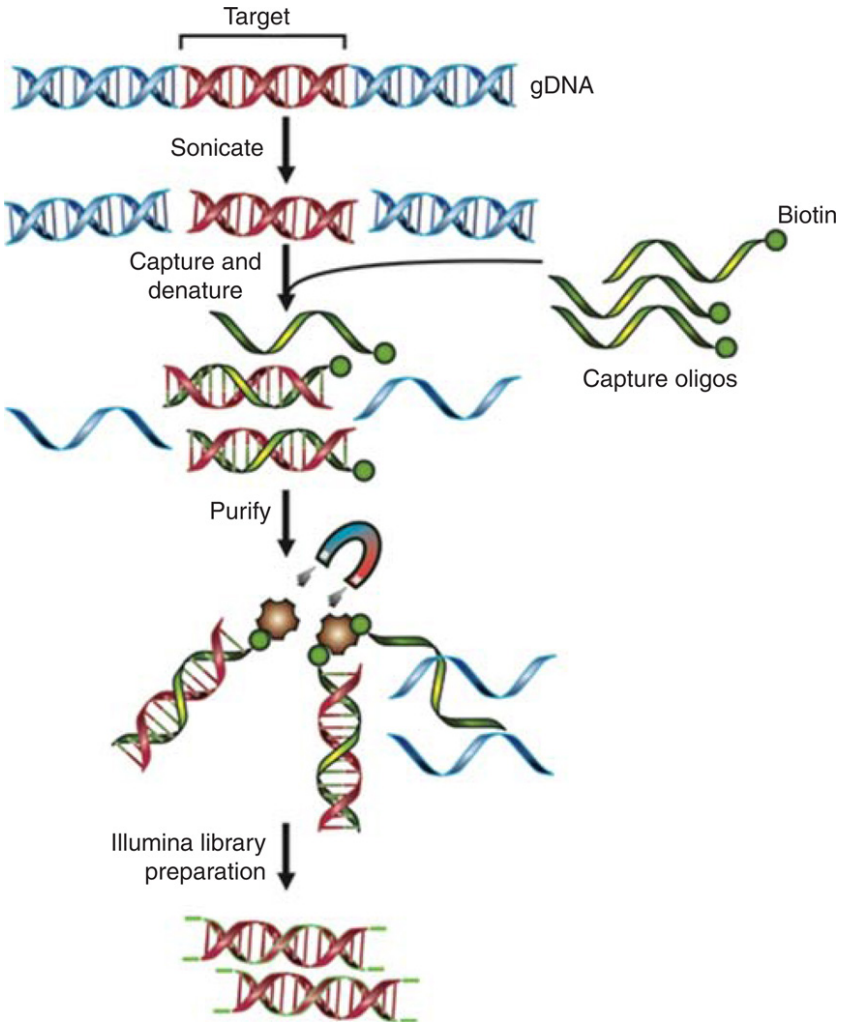

Figure 2: Whole Exome Sequencing Workflow adapted from [2]

1.8 Current Study

The primary goal of this project was to train machine learning models to identify and distinguish between true variants and artifacts. NGS errors were evaluated by using multiple Whole Exome Sequence datasets (WES) from the SRA database. The project aimed to utilize bioinformatics tools to identify these NGS artifactual variants, to train and test machine learning models for distinguishing between

these artifactual variants, and to validate the accuracy of the models by comparing the results with an NGS reference dataset. Additionally, this project aimed to investigate the predictive power of various features produced by variant calling software, to determine which sets of features are sufficient to identify artifacts in practice.

2 Methods and Materials

2.1 Bioinformatics Pipeline

Whole Exome sequencing (WES) datasets from NCBI were analyzed using the bioinformatics pipeline developed for this project as illustrated in Figure 3 to identify single nucleotide variants (SNVs). These SNVs were compared against the gold standard Genome in a Bottle (GIAB) truth set to determine false positive and false negative SNV calls [25] in human sequences.

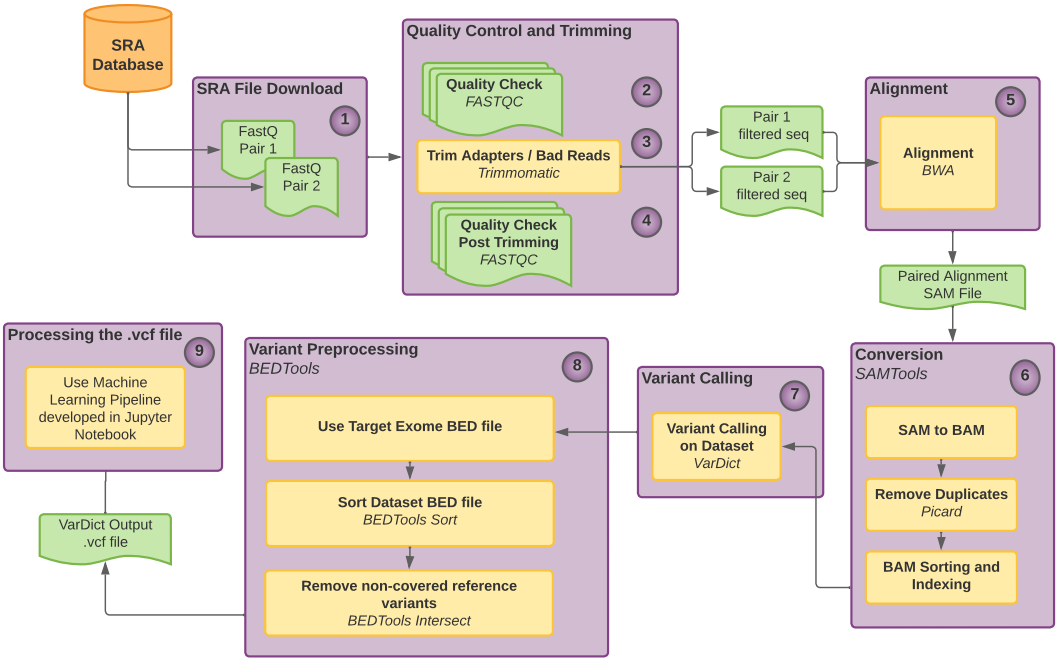

Figure 3: Overview of Bioinformatics Pipeline

2.1.1 Sequence Download and Conversion

The bioinformatics pipeline begins with downloading sequencing data from the SRA database by using the 'prefetch' function from SRA toolkit [26]. Sequence files were downloaded in the compressed SRA format, and then decompressed [19]. The raw sequencing data were then converted into the FASTQ file format using the 'fasterq-dump' function from SRA toolkit [19]. FASTQ is a text-based format for storing nucleotide sequences and their corresponding quality scores.

2.1.2 Quality Control

The per base sequence quality of the data were then evaluated using FASTQC to highlight any potential problems in the reads [27]. Problems could either originate in the sequencer or in the starting genomic library materials. FASTQC is a tool developed by Simon Andrews of Babraham Bioinformatics and commonly used to provide an overview of quality control metrics for raw NGS data [27]. FASTQC's final output is a summary report that highlights any areas where the library appears unusual.

2.1.3 Adapter Trimming

Sequencing adapters were trimmed from the ends of sequencing reads using Trimmomatic, a tool developed by A. Bolger [28]. During the NGS library preparation step, adapter sequences are ligated to both ends of the DNA fragments of interest in order for the DNA to bind to the flow cell [29]. To recover the target DNA sequence, it is important to remove adapter sequences since they can interfere with the downstream analysis, such as alignment to reads [30] [31].

FASTQC was run again to ensure base call quality was satisfactory and that the adapters were removed.

An example of a per base sequence quality graph from a FASTQC report is shown in Figure 4. This FASTQC report was generated after Trimmomatic was run to remove adapter reads. The sequence quality report showed all good quality

base calls, except for a few base calls slightly dropping towards the end of the read. However, this was expected as the majority of the base call quality on most platforms degrade overtime as the run progresses.

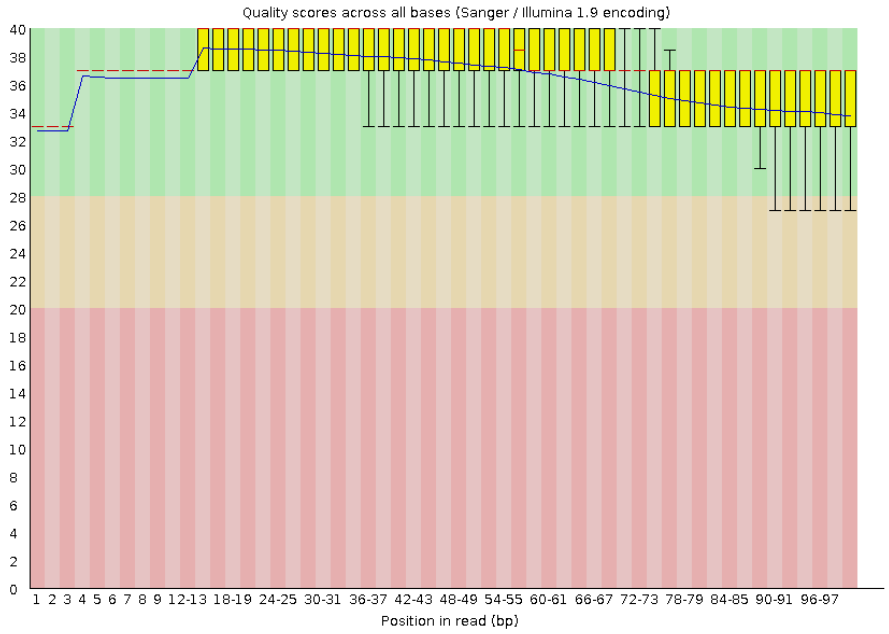

Figure 4: FASTQC Report Per base sequence quality for SRR2106342

2.1.4 Mapping to Reference Genome

The alignment step indicates which regions of the genome the read likely originated from [32]. This enables the reads to be assigned to a specific location in the genome. Each dataset used the hg38 assembly, Homo sapiens (human) genome assembly GRCh38, as the reference genome. During NGS runs, many short reads, or DNA fragments, are generated. These sequencing reads were aligned using the Burrows-Wheeler Aligner (BWA) [33] to hg38.

2.1.5 Removing Duplicates

After the alignment step, duplicate reads were removed using Picard's MarkDuplicates tool [34]. This tool removes PCR duplicate reads, which results in a more accurate read depth for the variants.

2.1.6 Variant Calling

Variant calling is the process of detecting locations where the reads differ from the reference genome with high confidence [35]. Variant calling was done using VardictJava [36], a versatile variant caller for DNA sequencing data. It compares aligned reads against the human reference genome, hg38. VardictJava calls SNVs, indels, and other complex variants and also detects differences in somatic and loss-of-heterozygosity variants between paired samples [37]. Variants are stored in VCF files. Any differences between the reference genome and the newly constructed sequence are labeled as variants.

Allele frequency refers to the frequency of a gene variant in a population [38]. The allele frequency threshold passed to VarDictJava was 0.01. This low value was chosen so that true variants and sequencing artifacts are included in the final VCF output, which would be used for training models.

2.1.7 Variant Filtering

Several variant filtering steps were implemented to ensure that variants called from VarDictJava referred to the appropriate high-confidence regions and exome regions. The diagram in Figure 5 illustrates the overlapping regions between SNVs identified from VarDictJava or GIAB high-confidence regions and the specific target exome capture region.

Many labs and universities use capture kits to perform whole exome sequencing. Because these various capture kits all contain their own specifications, a BED file is critical for specifying the region of interest. The BED file contains regions that indicate where the alignments are expected based on the specific target capture kit [39]. Using specific target exome BED files is essential as there are many different types of exome sequencing solutions, which can range wildly in size and exclusion of various regions. Using the correct exome bed file associated with each dataset ensures that the compared variants from the dataset and GIAB come from the same regions.

A package called BEDTools developed by A. Quinlan was used to identify the intersection between variants identified by VarDict in the SRA data and the target exome BED file, or GIAB high-confidence regions against the target exome BED file. BEDTools incorporates the genome-binning algorithm used by UCSC Genome Browser [40]. The genome-binning algorithm uses clustering approaches to group reads or contigs into "bins" that will then be assigned to a genome [40]. This feature accelerates the search for overlapping features [41].

GIAB has identified a set of "high-confidence" variant calls and regions [42]. Outside the high-confidence regions, the accuracy of variant calling is likely to be lower, so benchmarking against high-confidence calls will likely have higher accuracy [42]. Variants from the SRA datasets and GIAB high-confidence variant calls were compared against the target exome BED file to match the appropriate exome regions.

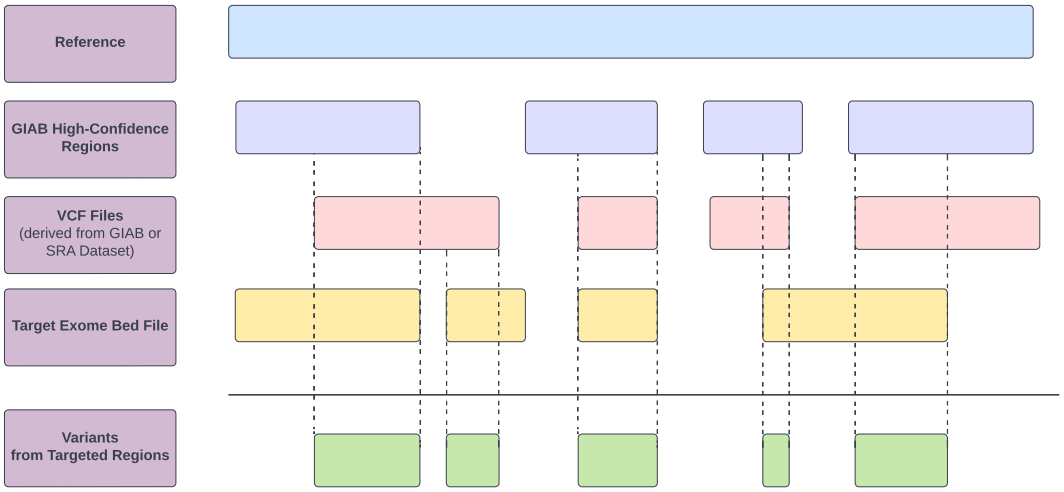

Figure 5: Identifying variants from targeted exome regions

Table 1 shows the appropriate target BED file was used for each SRA dataset. Both SRA record and the target BED file are hyperlinked in the table.

SRA Record ID	Exome Bed File Type
SRR14724463	Illumina TruSeq DNA Exome
SRR14724473	IDT Exome Targets
SRR14724493	Illumina TruSeq DNA Exome
SRR14724503	IDT Exome Targets
ERR1905890	Agilent SureSelect Human All Exon Kit V5
ERR1905889	Agilent SureSelect Human All Exon Kit V5
SRR2106344	Illumina Nextera Rapid Capture Target
SRR2106342	Agilent SureSelectv5 + UTR Target Enrichment
SRR1611184	SeqCap EZ Exome Capture Targets
SRR1611183	SeqCap EZ Exome Capture Targets
SRR1611182	SeqCap EZ Exome Capture Targets
SRR1611181	SeqCap EZ Exome Capture Targets
SRR1611180	SeqCap EZ Exome Capture Targets
SRR1611179	SeqCap EZ Exome Capture Targets
SRR1611178	SeqCap EZ Exome Capture Targets
SRR504510	ARUP SeqCap EZ Exome

Table 1: SRA Record and their Exome Targeted Regions

2.2 Artifact Identification

2.2.1 Differentiating between Artifact Types

The resulting dataset with all the SRA datasets combined from Table 1 contains the intersected variant calls from GIAB high-confidence variants, and the sequence dataset from the SRA record. The pipeline for this project categorized the different types of artifacts by merging the GIAB and SRA VCF files and indexing on 'CHROM', chromosome number, and 'POS', position. The sequencing artifacts were categorized as follows:

1. Case 1: Matching variants called in both the SRA .vcf file and the GIAB .vcf file at the same locus were considered a true variant and assumed to not be artifacts.
2. Case 2: Different variants called in both the SRA .vcf file and the GIAB .vcf file at the same locus were assumed to be artifacts.
3. Case 3: Variants in the GIAB .vcf file not called in SRA .vcf files were assumed to be artifacts, but were not modeled in this project so far. The reason that this group of artifacts are not modeled in this project will be discussed in the discussion section of this report.
4. Case 4: SRA variant calls not present in the GIAB .vcf file were also assumed to be artifacts.

The four types of artifacts are summarized in Table 2.

Table 2: Description of Artifact Types

GIAB	**Sequence Dataset**	**Label as:**
Variant Present	Variant Present	FALSE (not an artifact)
Variant Present	Different Variant Present in the same locus as GIAB	TRUE (an artifact)
Variant Present	Variant Not Present	TRUE (an artifact)
Variant Not Present	Variant Present	TRUE (an artifact)

The counts of each artifact case type were reported in Table 3 below for each of the SRA runs.

		Artifacts Statistics for each SRA Run				
		Case 1	Case 2	Case 3	Case 4	Total No. of Artifacts
SRA Runs	SRR14724463	24259	2	1601	102031	127893
	SRR14724473	16129	0	1295	55813	73237
	SRR14724493	24151	1	1740	89054	114946
	SRR14724503	16093	1	1330	58612	76036
	ERR1905890	17035	2	1124	37654	55815
	ERR1905889	17087	0	1074	43938	62099
	SRR2106344	24236	0	1626	240189	266051
	SRR2106342	49106	1	2595	95916	147618
	SRR1611184	38522	0	3031	46890	88443
	SRR1611183	38834	1	2718	52682	94235
	SRR1611182	38789	1	2763	55423	96976
	SRR1611181	38794	1	2758	52300	93853
	SRR1611180	38347	3	3203	48597	90150
	SRR1611179	38831	1	2721	53396	94949
	SRR1611178	38838	0	2715	53545	95098
	SRR504510	16730	0	2067	38082	56879

Table 3: Counts for each Artifact Case Type

The average counts of each artifact type are displayed in Figure 6.

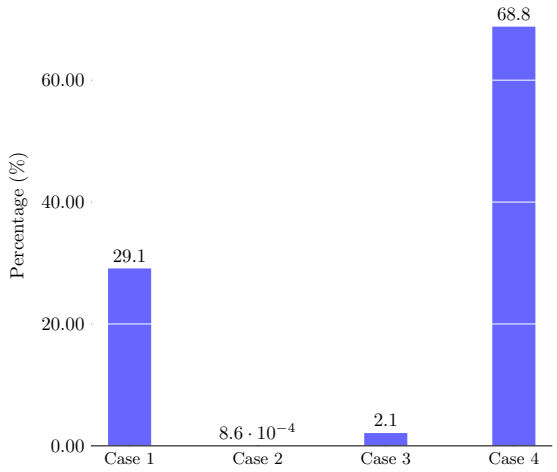

Figure 6: Percentages of each type of variant in the training data

2.2.2 Artifact Investigation

The heatmap shown in Figure 7 displays the number of SNV substitutions between 'REF', the nucleotide from the reference genome, and 'ALT', the variant nucleotide, among the artifacts in the training data. It shows which bases are most likely to be switched to other bases.

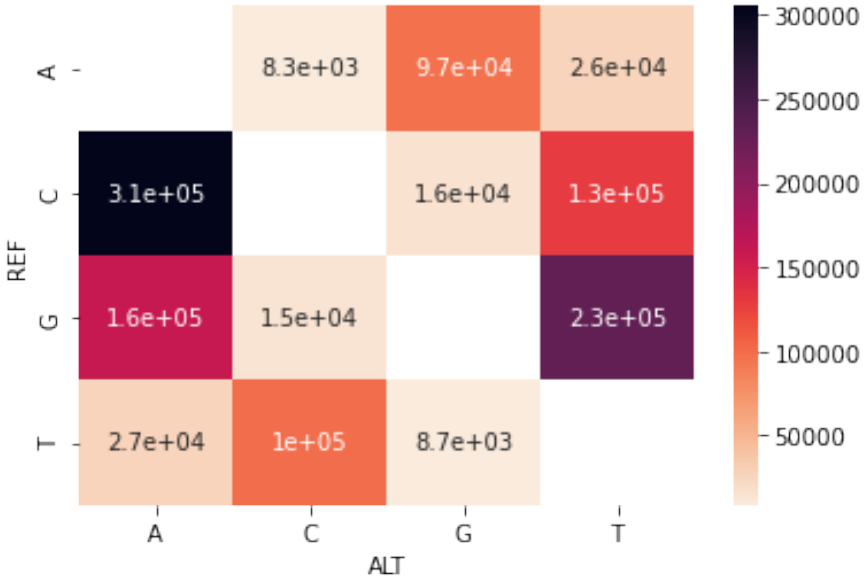

Figure 7: Frequencies of Sequencing Artifacts

2.3 Machine Learning Pipeline

Figure 8 contains an illustrated overview of the machine learning pipeline. After artifacts were identified and filtered through the bioinformatics pipeline they were imported into a Python notebook for data exploration. Following that, machine learning models were built to classify between true variants and artifacts. Major libraries used included Pandas [43] for data processing and exploration, Matplotlib [44] and Seaborn [45] for visualization, and Scikit-Learn [46] for machine learning and preprocessing.

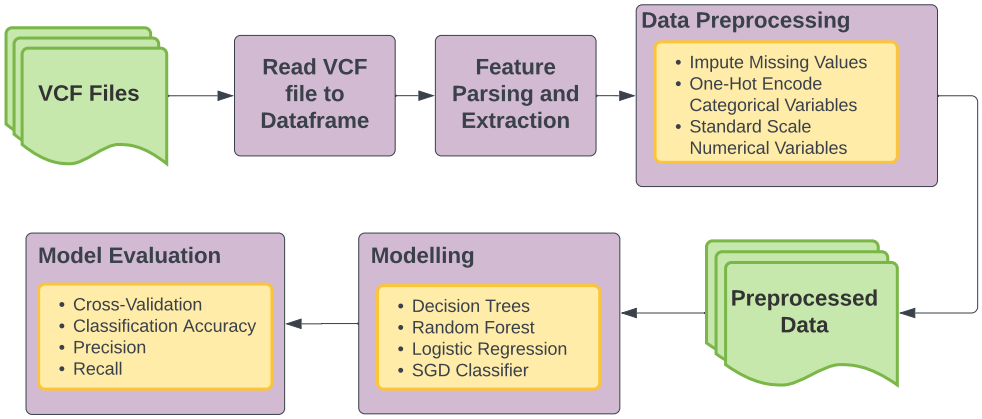

Figure 8: Overview of Machine Learning Pipeline

2.3.1 Feature Extraction

VCF files were parsed into a Pandas dataframe to be used in the machine learning pipeline. This was done by reading each VCF file as a tab-delimited file. Because VarDict contained extra data about each variant, it was necessary to split up the 'INFO' column into several sub-columns. Variant files associated with GIAB and SRR respectively were parsed. After parsing was complete, each dataframe was filtered to only include SNVs. Indels and other multi-nucleotide variants (MNVs) were excluded.

2.3.2 Data Exploration and Feature Selection

Data exploration was performed to provide insights on the dataset, which could then inspire subsequent feature selection for the model. Selecting good features is critical for modeling the data. For instance, it could potentially reduce overfitting, improve accuracy, and reduce training time.

All the categorical and numerical features were examined from the VarDict .vcf file and are found in the Appendix, in Table 11.

Plots were also made to visualize the data. To determine which features should be included, histograms and bar graphs were plotted to identify the data distribu-

tion and other patterns. Some features, such as AF (allele frequency) and HIAF (high quality bases allele frequency) appear to have good separation. Others did not appear to obviously separate the data, some had no variance at all, or some contained mostly null values.

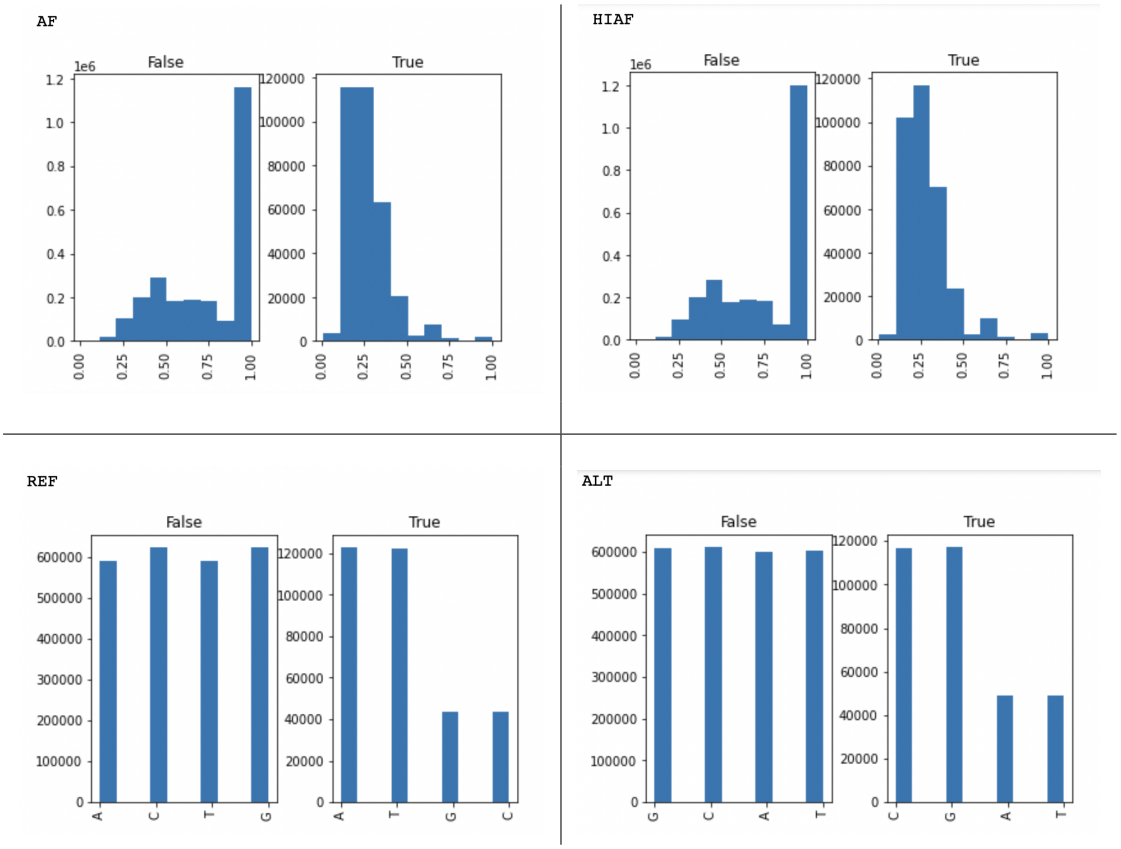

Figure 9: Histograms of Selected Features

2.3.3 Preprocessing prior to Modeling

After feature selection was completed, these were the features used in the model. Table 11 in the Appendix provides the descriptions of these features in more detail.

Table 4: Features Used in the Model

Type of Feature	**Names of Features**
Categorical	ALT, REF, BIAS, FORMAT, LongMSI, MSI12, NM5.25, PASS, Q10, REFBIAS, VARBIAS, p8, pSTD
Numerical	ADJAF, AF, DP, DUPRATE, HIAF, HICNT, HICOV, MQ, MSI, MSILEN, NM, ODDRATIO, PMEAN, PSTD, QSTD, QUAL, SBF, SHIFT3, SN, SPANPAIR, SPLITREAD, VD

Many real world datasets contain missing values. However, most Machine Learning libraries presume all values are present. To mitigate this, missing values were usually imputed. However, imputation was not needed for this dataset because all features, except for 'SVLEN' and 'SVTYPE', had no missing values. 'SVLEN' and 'SVTYPE' were also discovered to have almost all missing data. Therefore, those two features were dropped from the dataset.

One-hot encoding was also used to translate categorical features into a numeric format compatible with the machine learning algorithms. For a categorical variable of k categories, one-hot encoding assigns each example a length-k vector of mostly zeroes, where a single element is set to one according to the category. This is necessary because many machine learning algorithms are only capable of handling numerical data.

`StandardScaler` was also applied to the numerical features. Standard scaling is used to make features of different magnitudes and variances more comparable to each other [47]. The data was transformed such that its distribution had a mean value of 0 and standard deviation of 1.

Both numerical and categorical pipelines were then joined to create the `X` features, and the artifact/no-artifact column was used as the `y`-labels.

2.3.4 Testing Machine Learning Methods

Decision Tree, Random forest, SGD classifier, and Logistic Regression models were used to model the data. These were picked because they perform the best on large-scale data [48]. Balanced class weighting was applied to all models in order to reduce bias towards one prediction over another. Variants belonging to Case 1 were given a label of '0' and artifacts belonging to Case 2 and 4 were given a label of '1'. Because artifacts were shown to be much more common than non-artifacts, a model that always chose "ARTIFACT" would be right most of the time. Therefore, it was necessary to weight artifacts lower than non-artifacts when training models.

All models were trained and evaluated using 3-fold cross-validation on the entire dataset. Accuracy, precision, and recall were calculated from cross-validation results.

3 Results

3.1 Data Exploration

Once all the dataframes from each SRA dataset were concatenated, the whole dataset was further evaluated for optimal feature selection. As shown previously in Figure 9, histograms were plotted to visually inspect which features were separable for training.

Once those features were evaluated from visually inspecting the histograms, all categorical and numerical features were also verified by compiling a list of the top important features. 1000 times, 3 random features were selected, and a random forest model was trained on the chosen features. The average accuracy of each model containing that feature was reported in Figure 10. Accuracy was also shown to decrease slightly each time a feature was removed. This figure informs us which features or attributes are the most important for the model.

As shown in Figure 10, 'AF' or allele frequency was the most important attribute. This could be verified since it was also the most separable feature from the histogram analysis. The top 4 features with approximately 98%-99% are 'AF', 'HIAF', 'HICNT', 'VARBIAS', and 'VD'. These results also corroborate with our histograms, since 'AF' and 'HICNT' showed highly separable features.

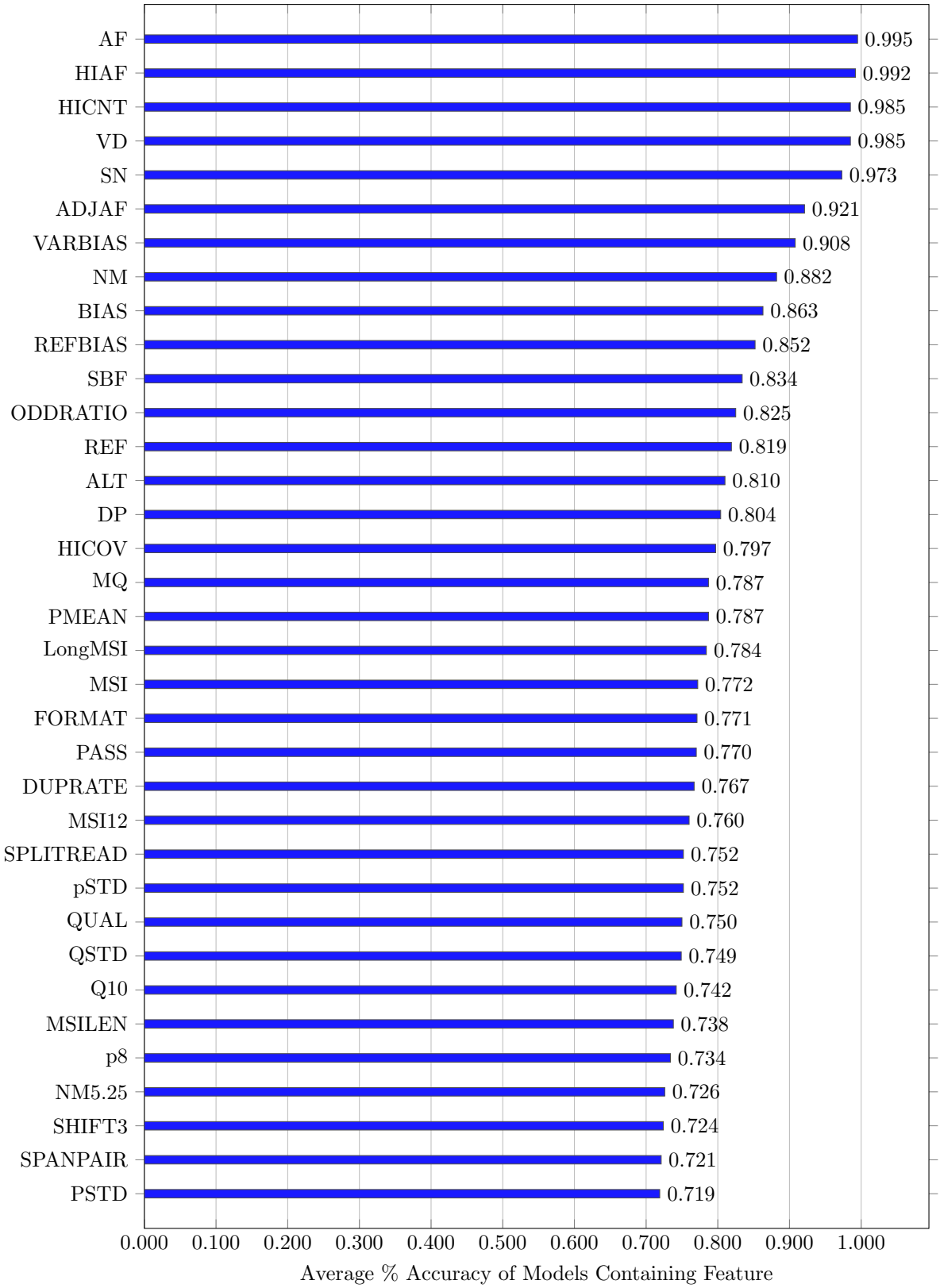

Figure 10: Relative Importance of Model Features

3.2 Performance Evaluation of Modeling Artifacts

3-fold cross-validation was used for assessing the models [49]. K=3 was chosen to ensure that each testing split had enough samples to accurately represent the distribution of the whole dataset. In addition, a higher number of folds would have led to unmanageably long computation time. Cross-validation was used to detect overfitting and other variances [49].

Confusion matrices were built using Scikit-Learn's packages. Classification metrics, such as precision and recall, and accuracy, can be calculated from a confusion matrix plot [50].

Precision and recall are useful indicators of prediction success, especially when classes are unbalanced. Precision measures the proportion of positive identifications that were actually correct. It is represented mathematically as the number of true positives over the total positive test instances [51]. Recall evaluates the proportion of actual positives that were identified correctly. It is represented mathematically as the number of true positives over the number of true positives plus false negatives [51]. A perfect classifier has precision and recall equal to 1.0.

In the context of this project, precision and recall are defined as:

$$\text{Precision} := \frac{\text{True, Predicted Artifacts}}{\text{Predicted Artifacts}} \quad \text{Recall} := \frac{\text{True, Predicted Artifacts}}{\text{True Artifacts}}$$

Lastly, ROC curves, AUC scores, and precision-recall curves were calculated and reported as well. The ROC curve is a probability curve and AUC represents the degree of separability between classes [52]. A high AUC score is indicative of how well the model can distinguish between classes. An AUC score of 1.0 means that the model is able to distinguish between classes perfectly. The ROC curve is plotted with the True Positive Rate (TPR) against the False Positive Rate (FPR). TPR and FPR are defined as:

$$\text{TPR} := \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}} \quad \text{FPR} := \frac{\text{False Positives}}{\text{True Negatives} + \text{False Positives}}$$

3.2.1 Decision Tree Model

A Decision Tree model resembles a tree structure, in which the nodes represent some threshold of an attribute and the branches represent the whether a given sample satisfies this threshold [53].

The code to compile a decision tree model is shown in the Appendix in Figure 21. The decision tree model had an accuracy of 99.7%, with a precision score of 99.9%, and a recall score of 99.6%. The confusion matrix is shown in Figure 11, along with the reported accuracy, precision, and recall scores in Table 5.

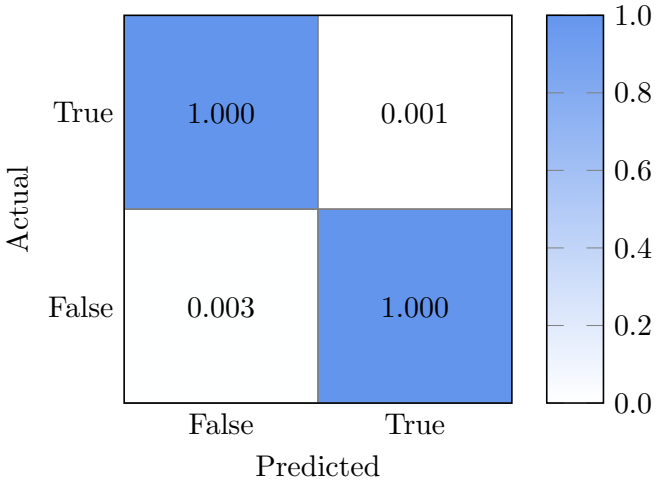

Figure 11: Decision Tree Confusion Matrix

Metric	Score
Cross-Validation Scores	99.7%
	99.7%
	99.7%
Accuracy	99.7%
Precision	99.9%
Recall	98.6%

Table 5: Decision Tree Classification Results

3.2.2 Random Forest Model

The Random Forest algorithm is composed of many small decision trees, each containing its own prediction. The model then uses averaging to combine all the

predictions from the decision trees into a more accurate prediction [54].

The code to compile a random forest model is shown in the Appendix in Figure 22. The random forest model had an accuracy of 98.7%, with a precision score of 99.8%, and a recall score of 98.8%. The confusion matrix is shown in Figure 12, along with the reported accuracy, precision, and recall scores in Table 6.

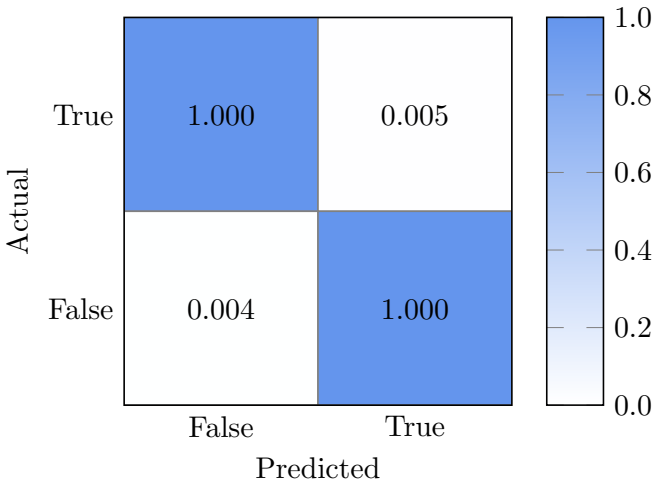

Figure 12: Random Forest Confusion Matrix

Metric	Score
Cross-Validation Scores	98.9%
	98.9%
	98.2%
Accuracy	98.7%
Precision	99.8%
Recall	98.8%

Table 6: Random Forest Classification Results

3.2.3 Logistic Regression Model

The Logistic Regression algorithm is a binary classification algorithm that assigns weights to features. The sigmoid function is used to convert predicted values to probabilities [55].

The code to compile a logistic regression model is shown in the Appendix in Figure 23. The logistic regression model had an accuracy of 99.8%, with a precision score of 99.9% and a recall score of 99.8%. The confusion matrix is shown in Figure

13, along with the reported accuracy, precision, and recall scores in Table 7.

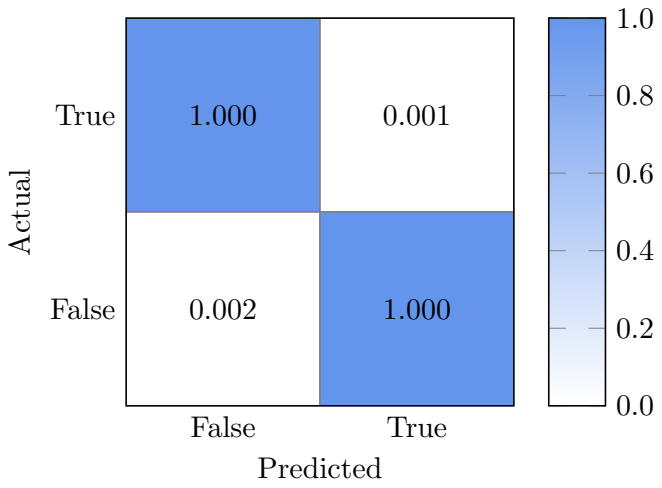

Figure 13: Logistic Regression Confusion Matrix

Metric	Score
Cross-Validation Scores	99.8%
	99.8%
	99.8%
Accuracy	98.8%
Precision	99.9%
Recall	99.8%

Table 7: Logistic Regression Classification Results

3.2.4 SGD Classifier Model

The SGD Classifier is another classification algorithm that minimizes loss of function using a fixed number of iterations, which makes it very fast to compute. It is known for operating on large datasets [56].

The code to compile an SGD classifier model is shown in the Appendix in Figure 24. The SGD Classifier model had an accuracy of 99.8%, with a precision score of 99.9% and a recall score of 99.8%. The confusion matrix is shown in Figure 14, along with the reported accuracy, precision, and recall scores in Table 8.

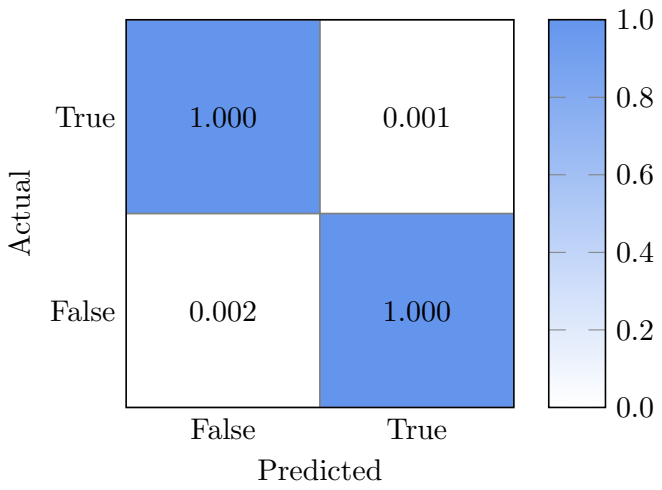

Figure 14: SGD Classifier Confusion Matrix

Metric	Score
Cross-Validation Scores	99.8%
	99.8%
	99.8%
Cross-Validation Mean Score	99.8%
Precision	99.9%
Recall	99.8%

Table 8: SGD Classifier Classification Results

3.2.5 Summary

A summary of each model's results are listed in Table 9.

		Classification Statistics for each Model		
		Precision	Recall	Accuracy
Model	Decision Tree	99.7%	99.9%	99.6%
	Random Forest	98.6%	99.8%	98.6%
	Logistic Regression	99.8%	99.9%	99.8%
	SGD Classifier	99.8%	99.9%	99.8%

Table 9: Classification Modeling Summary Results

3.2.6 ROC Curve and Precision-Recall Curve

The ROC curve comparison for logistic regression, random forest, and SGD classifier is shown in Figure 15. The ROC curve is a probability curve for binary classification problems and plots True Positive Rate (TPR) against False Posi-

tive Rate (FPR) [57]. The area under the curve (AUC) measures the ability of a classifier to distinguish between classes [57]. The higher the AUC value, the more robust the model is at predicting between classes. If the AUC value is 1, the classifier is able to distinguish between classes perfectly.

The Precision-Recall curve shows the trade-off between precision and recall. The closer the precision and recall values are to 1.0, the higher the accuracy is [51]. High precision scores indicate that the classifier is returning more accurate results and high recall scores indicate that the classifier is returning positive results as a majority [51].

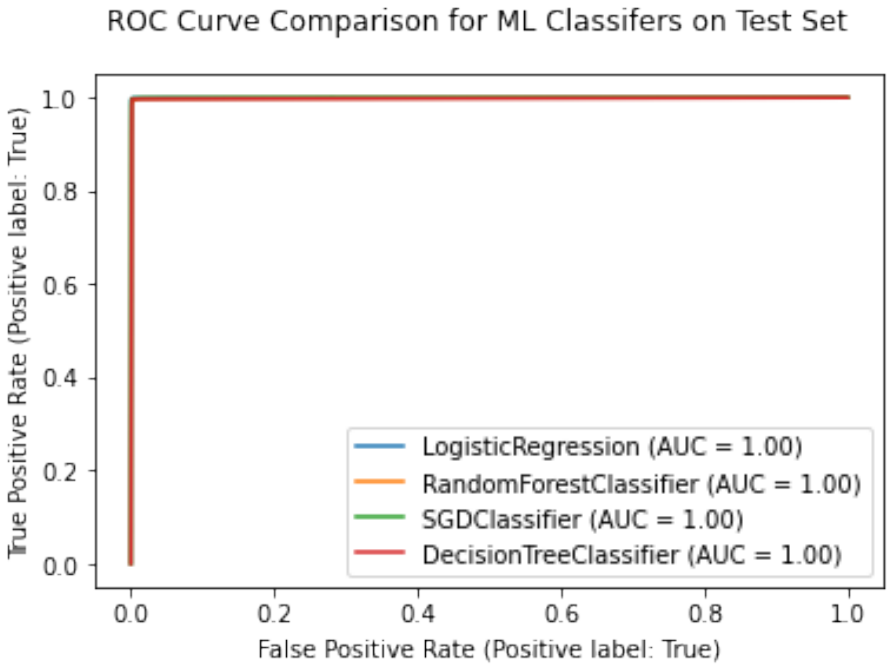

Figure 15: ROC Curve Comparison

All 4 curves have an AUC of 1.00, indicating that all 3 models are able to accurately predict whether a variant is an artifact or a non-artifact.

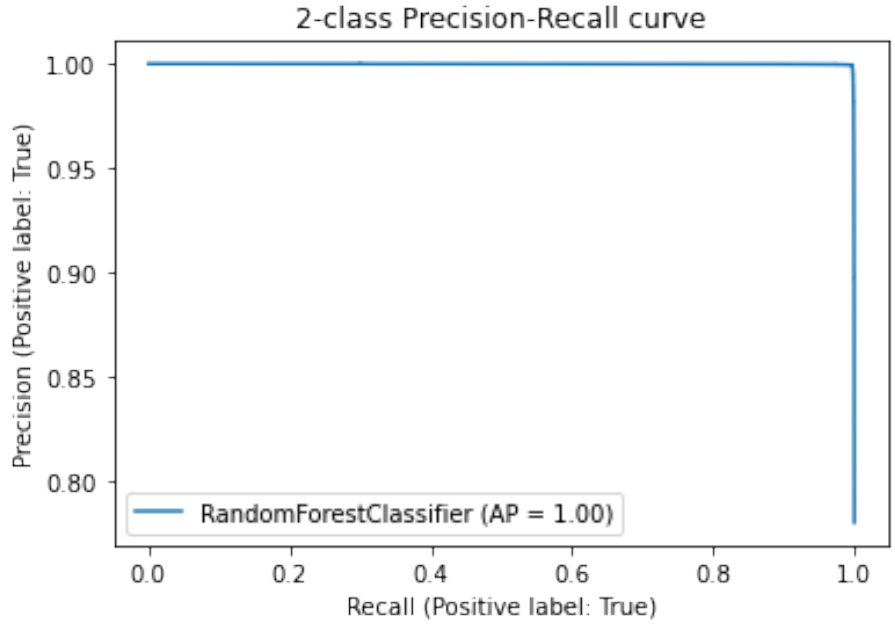

Figure 16: Precision-Recall Curve

The random forest model shows a high precision and a high recall value, indicating that the results are highly accurate.

3.3 Experiments

3.3.1 Feature Removal

The high performance of the models prompted further investigation into feature importance, to determine which features contributed the most predictive power. To test the predictive ability of each feature, every feature was removed in sequence, in order of calculated feature importance, to see how well the random forest model can predict between classes. As shown in Figure 17 and Table 10, accuracy, precision, recall, still remain high and only decreases slightly, from 98.7% to 96.7%. Removing the top five features do not have a significant impact on the model performance. However, as more features are removed, accuracy and recall decrease while precision remains high. This phenomenon is also reflected in Figure 17. The curves of each model trend upward more than they trend to the right. The results showed that a combination of less-predictive features can still

differentiate artifacts with accuracy of 75% to 90%.

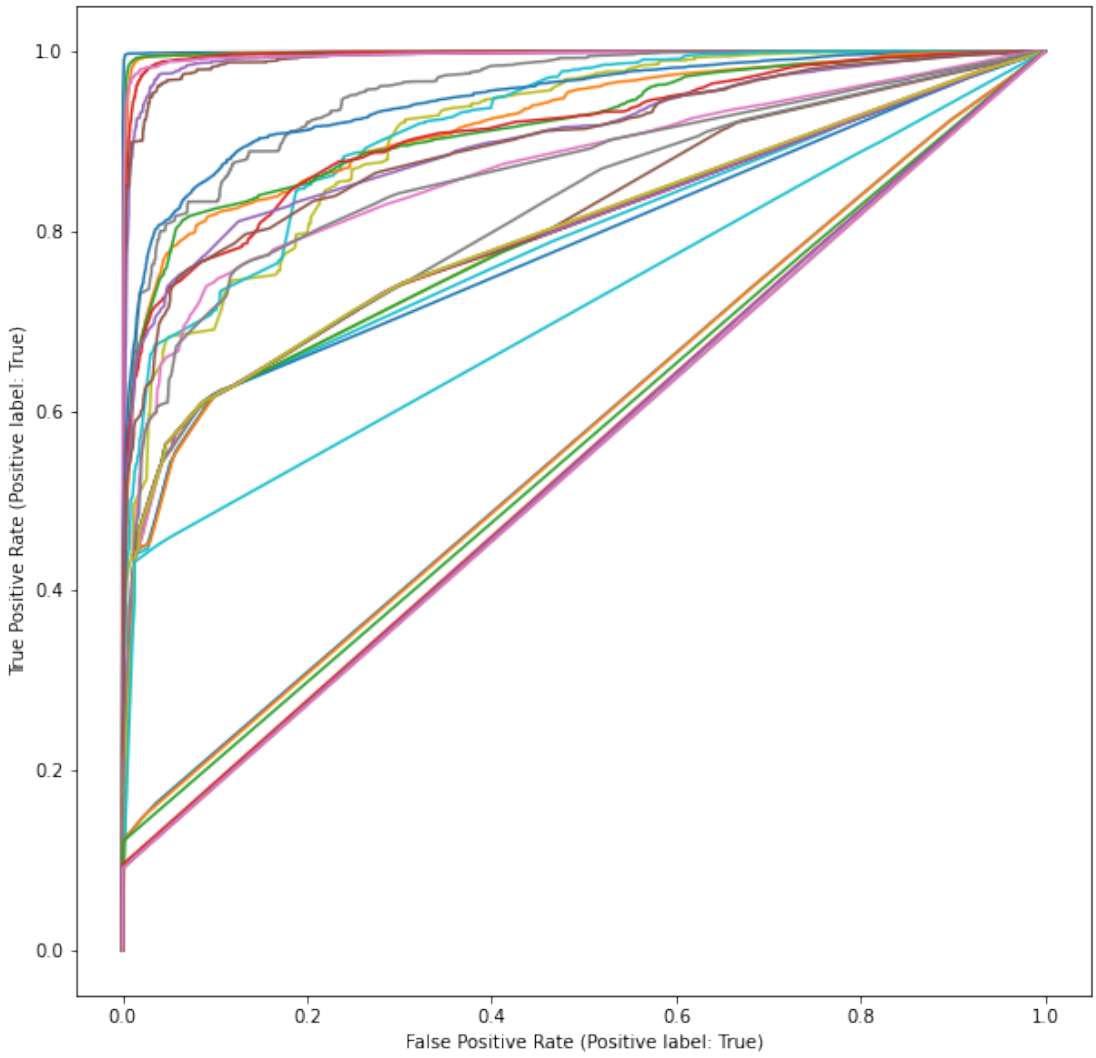

Figure 17: ROC Curve Comparison for Feature Removal

3.3.2 Correlation Among High Performing Features

Previous experiments led to the discovery that 'AF', 'HIAF', 'VD', and 'HICNT' are each able to separate artifacts with a high degree of accuracy. A correlation was plotted for these four features in order to understand whether these features all directly or indirectly coded for the same thing. Figure 18 is a pair plot that contains two figures, the histogram and the scatter plot. The histogram on the

Number Feature Set	Accuracy	Precision	Recall
All Features	98.7%	99.7%	99.8%
Previous Row Minus 'AF'	98.8%	99.8%	99.6%
Previous Row Minus 'HIAF'	98.2%	99.7%	99.1%
Previous Row Minus 'VD'	96.7%	99.3%	94.9%
Previous Row Minus 'HICNT'	94.0%	99.3%	93.0%
Previous Row Minus 'SN'	89.9%	99.1%	86.2%
Previous Row Minus 'ADJAF'	86.8%	98.9%	83.0%
Previous Row Minus 'PMEAN'	82.6%	99.3%	82.3%
Previous Row Minus 'QUAL'	83.9%	98.9%	80.5%
Previous Row Minus 'VARBIAS'	79.8%	95.8%	76.5%
Previous Row Minus 'NM'	78.1%	98.6%	67.8%
Previous Row Minus 'BIAS'	78.1%	98.5%	65.2%
Previous Row Minus 'HICOV'	70.2%	97.9%	72.5%
Previous Row Minus 'DP'	74.8%	98.2%	62.8%
Previous Row Minus 'REFBIAS'	77.4%	98.1%	75.3%
Previous Row Minus 'SBF'	81.9%	97.4%	80.5%
Previous Row Minus 'ALT'	76.3%	96.9%	74.2%
Previous Row Minus 'REF'	77.1%	96.7%	73.5%
Previous Row Minus 'PSTD'	77.7%	96.4%	74.6%
Previous Row Minus 'SHIFT3'	77.8%	96.2%	75.4%
Previous Row Minus 'SVTYPE'	77.0%	96.3%	75.1%
Previous Row Minus 'SPANPAIR'	77.7%	96.2%	75.3%
Previous Row Minus 'MQ'	77.8%	96.2%	75.3%
Previous Row Minus 'SPLITREAD'	77.7%	96.2%	75.2%
Previous Row Minus 'MSILEN'	77.9%	96.1%	75.5%
Previous Row Minus 'ODDRATIO'	60.3%	98.7%	50.4%
Previous Row Minus 'MSI12'	60.3%	98.7%	50.4%
Previous Row Minus 'QSTD'	45.5%	98.2%	31.3%
Previous Row Minus 'MSI'	45.5%	98.2%	31.3%
Previous Row Minus 'PASS'	45.5%	98.2%	31.2%
Previous Row Minus 'SVLEN'	45.5%	98.2%	31.2%
Previous Row Minus 'Q10'	43.1%	99.7%	27.6%
Previous Row Minus 'pSTD'	24.8%	98.1%	3.9%
Previous Row Minus 'FORMAT'	24.8%	98.1%	3.9%
Previous Row Minus 'p8'	22.3%	93.8%	0.6%

Table 10: Feature Removal Statistics

diagonal shows the distribution of a single variable, while the scatter plots show the relationship between the two variables. Orange points are artifacts and blue points are true variants.

It can be seen that 'AF' and 'HIAF' show a positive correlation, as seen in the second graph on the top row, and the graph on the second row, in the far left position. They are very similar since they almost align with each other. All the false artifacts also cluster together in lower 'AF' value range, from 0 to approximately 0.3. It can also be seen that 'VD' and 'HICNT' show a strong positive correlation, as seen in the graph on the third row, far right, and the third graph on the bottom row.

The other features do not show any obvious correlations. They do follow the trend that false artifacts tend to be associated with low values of the feature.

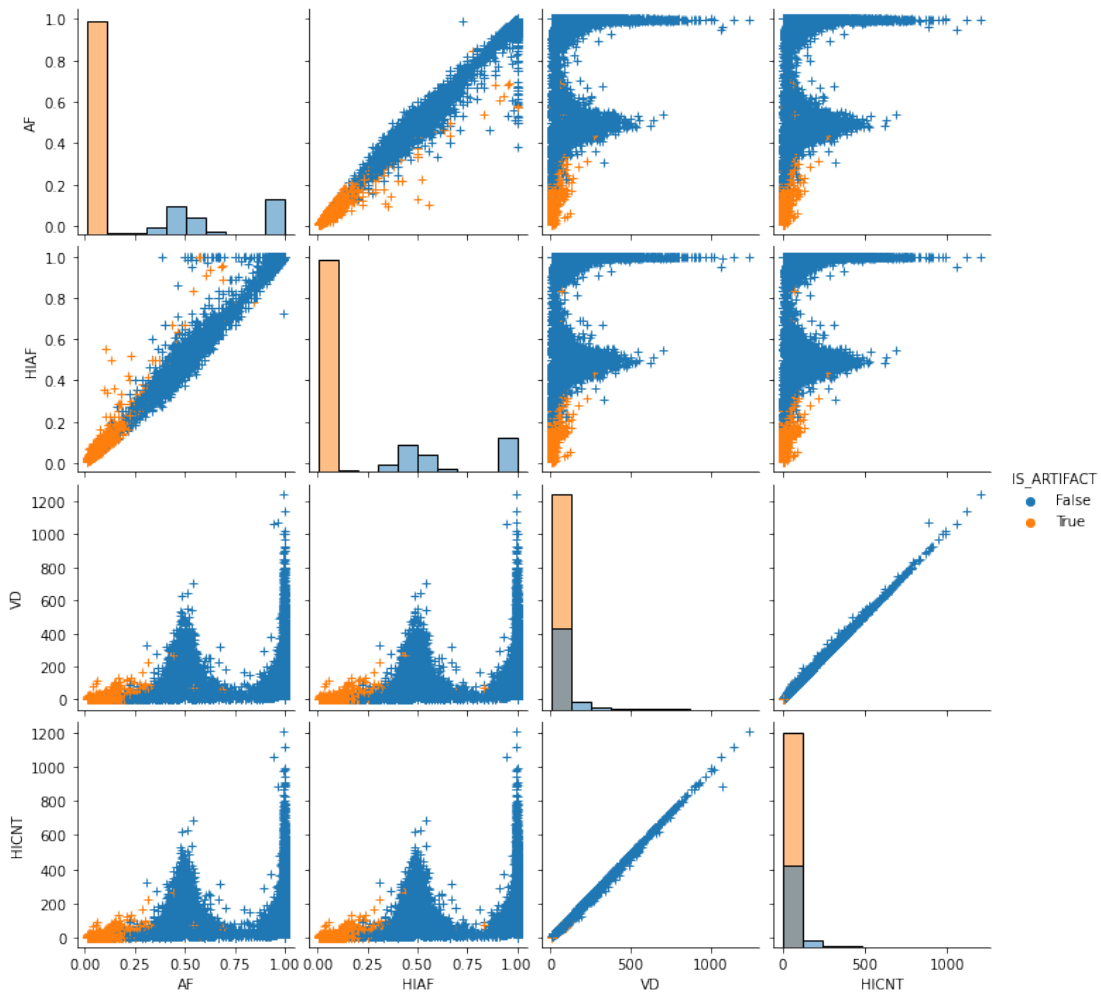

Figure 18: Correlation Plot between Features ‘AF’, ‘HIAF’, ‘VD’, and ‘HICNT’

3.3.3 Determining Optimal AF Threshold

Because 'AF' was discovered to be the most predictive feature, and it is also an important parameter for variant callers, graphs were plotted to determine an optimal range of 'AF' values.

To determine optimal 'AF' values for distinguishing artifacts, the following rule was used: "For a given threshold t, classify a variant as a non-artifact if it has 'AF' greater than t". This rule produces a model where accuracy is determined by the percentage of correctly-classified variants, and where the 'positive' outcome in precision and recall calculations is a non-artifactual variant. It is worth noting that this is the opposite of the terminology used in the main model-training task, where artifacts are the positive class.

To identify regions of interest, the value of t was varied between 0 and 1, and accuracy, precision, and recall were plotted as a function of t. These plots can be seen in Figure 19 and Figure 20. The value of t that optimizes classification of artifacts is 0.224 (shown with a dotted red line). The accuracy is at least 99% for values of t between 0.1 and 0.35. It can also be seen that precision (the fraction of non-artifacts in the output) is very low before $t = 0.1$, and that the recall (the fraction of not artifacts that make it to the output) falls off sharply after $t = 0.35$. There is also a plateau in recall for values of t between 06 and 0.95.

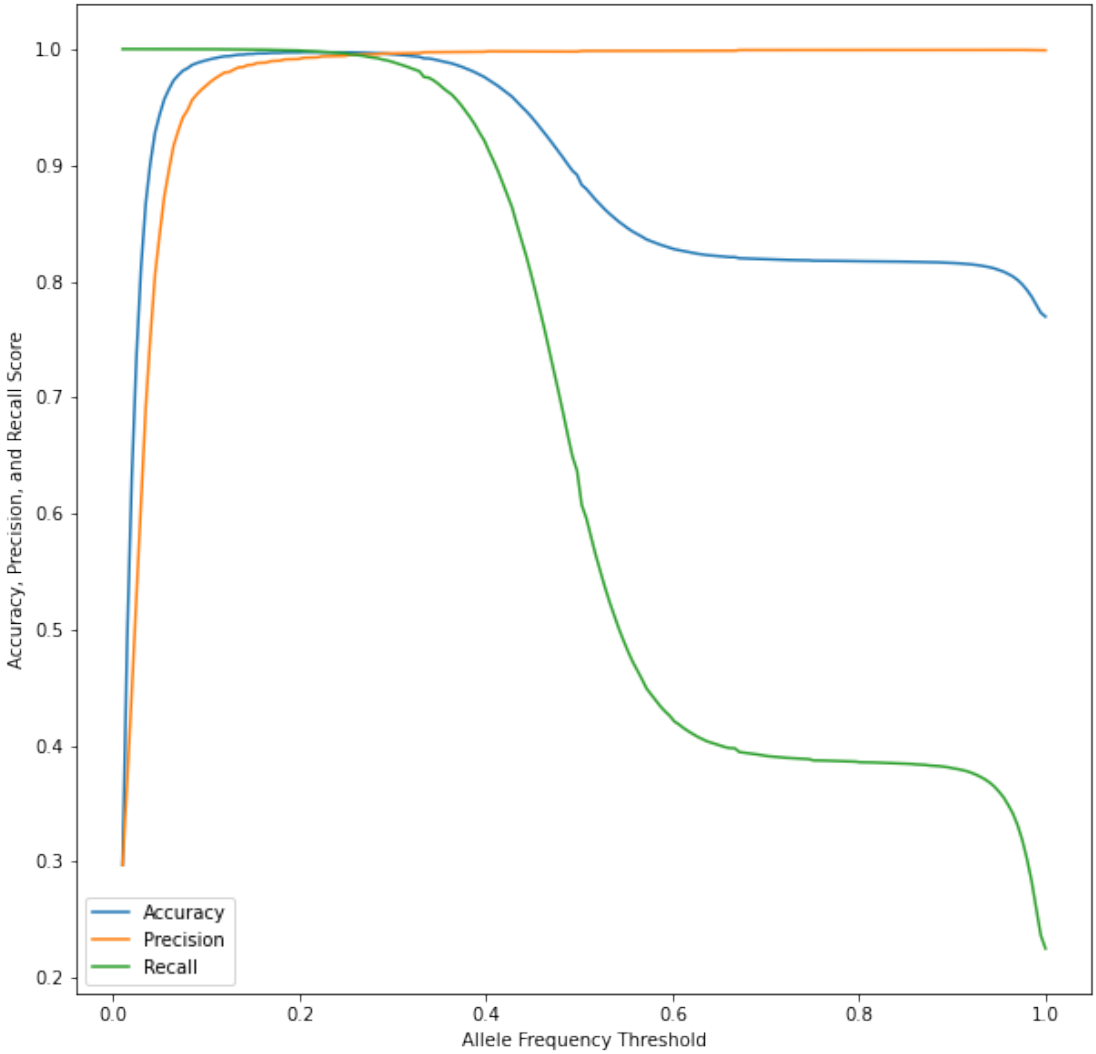

Figure 19: Accuracy, Precision, and Recall Curves for AF

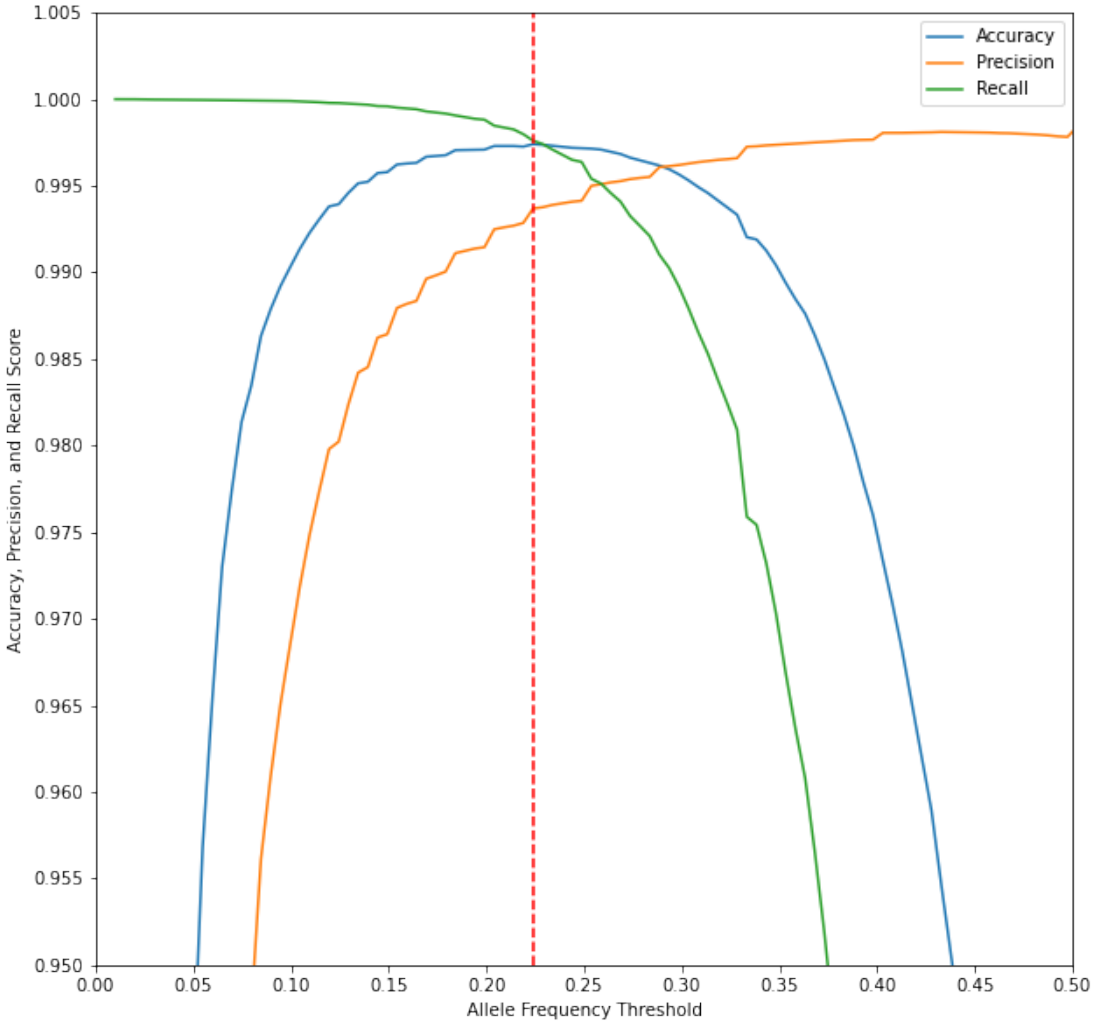

Figure 20: Optimal AF Threshold

4 Discussion

4.1 Summary and Conclusion

A total of 15 WES datasets were chosen from the SRA database as sources of artifacts. Each dataset was processed with the bioinformatics pipeline to produce sets of variants, some of which were artifacts introduced during the NGS process. These variants files were compared against ground-truth variants from GIAB to produce a labeled training set of true and artifactual variants. Machine learning models were then trained to differentiate between true and artifactual variants. All 4 models showed a high classification accuracy, averaging around 98%, as well as high precision and recall scores around 99%. Further experiments were conducted to analyze the impact of individual features on model accuracy. The most impactful feature was 'AF' (Allele Frequency), with it alone being able to correctly classify over 99% of the training set. Given that 'AF' is an important parameter used in variant callers, an investigation was conducted to recommend reasonable values. It was found that an 'AF' value of about 0.22 was best for distinguishing artifacts directly in the variant caller.

4.2 Artifact Investigation

The number of SNV substitutions among the artifacts in the training data between 'REF', the nucleotide from the reference genome, and 'ALT', the variant nucleotide, are shown in Figure 7. C>A and G>T base substitutions are the most prevalent, ranging from approximately 230,000 to 310,000 substitutions. These results follow similar findings as Costello, et al. [13]. Costello, et al. discovered that C>A and G>T base substitutions are typically found in low allelic fractions in targeted capture data [13]. These artifacts were possibly introduced through DNA shearing protocols that are a result of oxidation in DNA [13].

4.3 Model Selection

Supervised learning algorithms were chosen for this project since labels were created to classify between artifacts and non-artifacts. The decision tree, random forest, logistic regression, and SGD classifier models were selected since they fall under the supervised learning category.

A decision tree uses a tree-like model that consists of nodes depicting the decisions and each decision's possible outcomes. Decision trees inherently performs feature selection and can handle both numerical and categorical data [58]. They also do not require much data preparation. However, decision trees have high variance as any small variations in the data will result in a different tree being generated [58]. At times, decision trees can create complex trees that do not generalize the data well, which can lead to overfitting as well [58].

Similarly, the random forest algorithm consists of multiple decision trees that work together to create a prediction [59]. Random forest often reduces overfitting and has high accuracy. Despite these advantages, random forest has some drawbacks. Random forest requires a lot of computational power since it is compiling many trees' outputs together [59]. Due to this larger amount of computational power, training time will also be increased.

Logistic regression is a classification algorithm used to describe the relationship between an independent variable and a binary dependent variable [55]. This model is often considered to be fast and easy to implement. For simpler data sets, the logistic regression algorithm performs well when the dataset is linearly separable [55]. However, the model does not handle categorical variables as well it as it handles continuous variables. The number of observations and features should correlate, otherwise it can lead to overfitting.

Stochastic Gradient Descent (SGD) is another classification algorithm that is similar to the gradient descent. Instead of calculating the loss of function from all the data points, SGD calculates only the loss of function from one data point, which makes the algorithm operate much faster [56]. SGD is good for applying to

large-scale data, including training examples and features [56].

Due to the many advantages and disadvantages of each model, all 4 were selected and tested for comparison.

Though all models had similar performance on the main modelling task, it was decided that exploratory modelling would be done using the Random Forest model. It was chosen because it can be parameterized to control bias and variance [59]. It is also appropriate for modelling the relationships between features, as well as handling categorical data gracefully [59]. In contrast, simpler models like logistic regression may struggle to model the relationship between pairs of features [55].

4.4 Modeling Results Analysis

Review of the results showed that the models were able to accurately classify true and artifactual variants. All 4 models performed similarly, each achieving accuracy, precision, and recall in the 98%-100% range.

Care was taken to ensure the generalizability of the models, and to avoid overfitting. Towards this end, 3-fold cross validation was used. 3-fold was chosen as a trade-off between accuracy, computational power, and diversity in the testing split. With the k-fold cross-validation technique, the training data is split into k randomized subsets. For each of the k subsets, the model is trained on the other $k-1$ subsets before being tested on this subset. The mean accuracy score is considered more robust than that of a single train-test split [49], and the variance in the accuracy scores can reveal problems with generalization and overfitting.

It was found that for each model, there was very little variance in accuracy between each of the 3 cross-validation runs. This is evidence that the models generalize well, and are not subject to overfitting on the training set.

4.5 Feature Importance Survey

Given that the models were able to achieve high accuracy, an experiment was conducted to determine the impact of a feature on a model's ability to classify variants. 1000 separate Random Forest models were trained on randomized, size-3 subsets of the available features. Then, each feature was given an importance score that was the average accuracy of models trained on that feature.

This method for determining feature importance can be seen as a type of feature ablation study [60], where most features are removed from the model. It was known when designing this experiment that some features (such as 'AF' and 'HIAF') were highly correlated, and therefore removing just one of them was not likely to change model performance. It was decided to use 3 features per run to minimize the impact of these highly-correlated features.

4.6 High-Power Features

The results of the feature importance survey show that 4 of the features, 'AF', 'HIAF', 'VD', and 'HICNT' tend to increase the accuracy of any model training on them to at least 99%. This raises the question of why these features have such high predictive power, as well as whether these features are in some way correlated.

It can be explained that 'AF', 'HIAF', 'VD', and 'HICNT' have comparable model performance metrics because the features themselves could be similar in nature. Since they are similar, eliminating only one of them will not greatly affect the performance of the model. For instance, 'AF' and 'HIAF' represent 'allele frequency' and 'AF using only high quality bases'. It can be deduced that these 2 features contain similar data and perhaps 'HIAF' contains a more filtered version of 'AF'. 'VD' and 'HICNT' represent 'variant depth' and 'number of high-quality reads with the variant'. Since both are related to read coverage and depth, it can also be deduced that these 2 features contain similar data as well.

As seen in the results from Figure 17 and Table 10, 'AF', 'HIAF', 'VD', and 'HICNT' all display little to no decrease as each one of those features were sub-

sequently removed. It was confirmed in Figure 18 that these features did indeed have a strong correlation with each other. In the scatter plots of Figure 18,'AF' and 'HIAF' shows a positive correlation. 'VD' and 'HICNT' also show a strong positive correlation. Other combinations of these features do not show any relationship. These results confirm the hypothesis that these metrics perform similarly because they contain similar data.

Rare variants are classified as having an allele frequency of $< 5\%$ [61]. Since the likelihood of a true variant found with such a low allele frequency is so rare, it would make sense that the majority of false artifacts were found in Figure 18 fell in the lower spectrum of allele frequency.

4.7 Testing How Accuracy is Affected by Removing Features

The results of the feature importance survey showed that some features can greatly increase the accuracy of models that include them. This then leads to a question of whether a model is capable of identifying artifacts without these high-power features. To answer this question, another ablation study was run where features would be successively removed from the model in order of importance score.

Figure 10 shows which features had the most effect on the models. As discussed previously, 'AF', 'HIAF', 'VD', 'HICNT', each have high predictive power. The model accuracy only decreases slightly when they are removed, dropping from 98.7% to 96.7%. However, when all of them are removed, the accuracy drops to 94%. One interpretation of this is that the remaining features together have sufficient predictive power to achieve over 90% without the most important features.

As more features are removed, the accuracy score does not decrease monotonically. This may be attributed to the probabilistic nature of the Random Forest classifier. It can also be seen that precision tends to stay high, above 96%, even as many features are removed. However, it can also be seen that recall tends to decrease as more features are removed. This phenomenon can also be seen in Figure 17, where the ROC curves of each model tend to go upwards faster than

they go right.

One possible interpretation is that there are two types of artifactual variant - those that are easily identified and those that are clustered with non-artifactual variants. This situation would lead to high precision (due to the easily-distinguished artifacts), but low recall (due to the mixing of the variant types).

Finally, it can be seen that the models do not approach the accuracy of random classification (about 22%) until nearly all features have been ablated. This means that most features included in the model have some amount of predictive power.

4.8 AF Threshold Determination

The results of the feature importance survey suggest that 'AF' (Allele Frequency) is the most predictive feature for distinguishing between true and artifactual variants. This feature is also an important parameter that can be passed directly to variant callers such as Vardict-Java [36]. For that reason, an investigation was run to determine optimal values for 'AF' for use in variant calling.

Alleles represent an alternative form, or variant, of the gene that is present on the same genetic locus of the chromosome. Allele frequency refers to the frequency of a gene variant in a population [38] and is calculated by counting the number of times an allele is observed in the population and then dividing by the total number of copies of all the alleles at that specific genetic locus [38]. Allele frequencies show the genetic diversity of a population. Changes in allele frequency may indicate that new mutations have been introduced into the population [38].

It was found that an 'AF' value of about 0.22 was best for distinguishing artifacts directly in the variant caller. The results also suggest a range of appropriate threshold values for allele frequency, depending on the situation. For example, a researcher with a low tolerance for artifacts may use a value around 0.4, while one interested in catching every true variant may use a value around 0.1.

4.9 Future Research

Future research with incorporating other variant callers in the machine learning pipeline can be done. Currently, the machine learning pipeline in this project is optimized for VarDict specifically. It would be beneficial to make this pipeline robust to process VCF file formats from other variant callers as well. This could be done to create different parsing functions for each variant caller type that extracts the features of interest.

Currently, Case 3 artifacts were not included in the models since it is not possible to train on the absence of data in a model. It would be impossible to identify these from a VCF file. In order to identify these missing variants, models would need to be trained directly on the aligned reads rather than on called variants.

Another area worth exploring is investigating sequencing artifacts in other species, provided that they have a robust reference genome, such as the one from Genome in a Bottle.

To gain more confidence in the models' generalization and real-world predictive ability, several kinds of validation could be performed. For example, a series of experiments investigating the optimal number of k in n-fold cross validation can be performed to ensure higher confidence in accuracy results. Another validation strategy would be to perform cross validation by reserving entire datasets for testing.

Additionally, more research can be done in the curation of SRA datasets. This project has discovered the importance of using the appropriate target exome BED file for comparing expected regions of alignment between the reference dataset and the sample dataset. Further application of natural language processing (NLP) can be used for text classification in order to more easily find the correct text pertaining to the appropriate target exome BED file.

References

[1] iRepertoire, "NGS overview: From Sample to Sequencer to Results," https://irepertoire.com/ngs-overview-from-sample-to-sequencer-to-results/, 2022, accessed: 2022-03-29.

[2] R. Chen *et al.*, "Whole-exome enrichment with the agilent sureselect human all exon platform," *Europe PMC*, vol. 7, pp. 626–633, 2015.

[3] J.S. Mattick *et al.*, "The impact of genomics on the future of medicine and health," *Medical Journal of Australia*, vol. 201, pp. 17–20, 2014.

[4] Chris Townsend, "How Next-Generation Sequencing is poised to disrupt healthcare," https://www.wellspring.com/blog/how-next-generation-sequencing-is-poised-to-disrupt-healthcare, 2019, accessed: 2022-03-15.

[5] J. Adams *et al.*, "Sequencing human genome: the contributions of francis collins and craig venter," *Nature*, vol. 1, p. 133, 2008.

[6] A. Arteche-López *et al.*, "Sanger sequencing is no longer always necessary based on a single-center validation of 1109 ngs variants in 825 clinical exomes," *Scientific Reports*, vol. 11, 2021.

[7] S. Behjati *et al.*, "What is next generation sequencing," *BMJ Open Access*, vol. 98, pp. 236–238, 2013.

[8] Illumina, "Understanding the NGS workflow," https://www.illumina.com/science/technology/next-generation-sequencing/beginners/ngs-workflow.html, 2022, accessed: 2022-03-16.

[9] Genomics Education Programme, "Reference Genome: Defining Human Difference," https://www.genomicseducation.hee.nhs.uk/blog/reference-genome-defining-human-difference/, 2022, accessed: 2022-04-05.

[10] Medline Plus, "What are single nucleotide polymorphisms (SNPs)?" https://medlineplus.gov/genetics/understanding/genomicresearch/snp/#:~:text=SNPs%20occur%20normally%20throughout%20a,SNPs%20in%20a%20person's%20genome, 2022, accessed: 2022-03-17.

[11] N. Tanaka *et al.*, "Sequencing artifacts derived from a library preparation method using enzymatic fragmentation," *PLOS ONE*, vol. 15, 2020.

[12] H. Do *et al.*, "Sequence artifacts in dna from formalin-fixed tissues: Causes and strategies for minimization," *Clinical Chemistry*, vol. 61, pp. 64–71, 2015.

[13] M. Costello *et al.*, "Discovery and characterization of artifactual mutations in deep coverage targeted capture sequencing data due to oxidative dna damage during sample preparation," *Nucleic Acids Res*, vol. 41, 2013.

[14] Q. Peng *et al.*, "Reducing amplification artifacts in high multiplex amplicon sequencing by using molecular barcodes," *BMC Genomics*, vol. 589, 2015.

[15] NIST, "Genome in a Bottle," https://www.nist.gov/programs-projects/genome-bottle, 2022, accessed: 2022-03-18.

[16] Genome in a Bottle *et al.*, "Genome in a bottle—a human dna standard," *Nature*, vol. 33, 2015.

[17] M. Cheung *et al.*, "Systematic bias in high-throughput sequencing data and its correction by beads," *Nucleic Acids Research*, vol. 39, 2011.

[18] National Institute of Health, "About the Sequence Read Archive (SRA)," https://datascience.nih.gov/data-ecosystem/sra, 2020, accessed: 2022-03-18.

[19] SRA, "Download SRA sequences from Entrez search results," https://www.ncbi.nlm.nih.gov/sra/docs/sradownload/, 2021, accessed: 2021-12-06.

[20] Healio, "Whole Genome Sequencing vs. Whole Exome Sequencing vs. Target Sequencing Panels," https://www.healio.com/hematology-oncology/learn-genomics/whole-genome-sequencing/whole-genome-whole-exome-sequencing-and-targeted-sequencing-panels, 2020, accessed: 2022-03-21.

[21] Y. Barbitoff *et al.*, "Systematic dissection of biases in whole-exome and whole-genome sequencing reveals major determinants of coding sequence coverage," *Nature: Scientific Reports*, vol. 10, 2020.

[22] CD Genomics Inc., "Principles and Workflow of Whole Exome Sequencing," https://www.cd-genomics.com/resourse-Principles-and-Workflow-of-Whole-Exome-Sequencing.html, 2022, accessed: 2022-03-31.

[23] ClinVar NCBI, "What is ClinVar?" https://www.ncbi.nlm.nih.gov/clinvar/intro/, 2022, accessed: 2022-04-07.

[24] P. Dunn *et al.*, "Next generation sequencing methods for diagnosis of epilepsy syndromes," *Frontiers in Genetics*, vol. 9, 2018.

[25] J. Zook *et al.*, "An open resource for accurately benchmarking small variant and reference calls," *Nature Biotechnology*, vol. 37, pp. 561–566, 2019.

[26] NCBI SRA, "NCBI SRA Tools," https://github.com/ncbi/sra-tools, 2022, accessed: 2022-04-07.

[27] S. Andrews, "FASTQC," https://www.bioinformatics.babraham.ac.uk/projects/fastqc/, 2021, accessed: 2021-12-06.

[28] A. Bolger *et al.*, "Trimmomatic: A flexible trimmer for illumina sequence data," *Bioinformatics*, vol. 30, no. 15, pp. 2114–2120, 2014.

[29] S. Head *et al.*, "Library construction for next-generation sequencing: Overviews and challenges," *Biotechniques*, vol. 56, no. 2, pp. 1–31, 2014.

[30] H. Jiang *et al.*, "Skewer: a fast and accurate adapter trimmer for next-generation sequencing paired-end reads," *BMC Bioinformatics*, vol. 15, no. 182, pp. 1–12, 2014.

[31] Illumina, "Adapter trimming: Why are adapter sequences trimmed from only the 3' ends of reads?" https://support.illumina.com/bulletins/2016/04/adapter-trimming-why-are-adapter-sequences-trimmed-from-only-the--ends-of-reads.html, 2021, accessed: 2021-12-06.

[32] K. Howe *et al.*, "The zebrafish genome sequencing project: Bioinformatics resources," *Nature*, vol. 496, pp. 498–503, 2013.

[33] H. Li *et al.*, "Fast and accurate short read alignment with burrows-wheeler transform," *Bioinformatics*, vol. 25, pp. 1754–60, 2009.

[34] GATK, "MarkDuplicates (Picard)," https://gatk.broadinstitute.org/hc/en-us/articles/360037052812-MarkDuplicates-Picard-, 2022, accessed: 2022-04-08.

[35] Z. Bohannan *et al.*, "Calling variants in the clinic: Informed variant calling decisions based on biological, clinical, and laboratory variables," *Computational and Structural Biotechnology Journal*, vol. 17, pp. 516–569, 2019.

[36] Z. Li, "VarDictJava," https://github.com/AstraZeneca-NGS/VarDictJava, 2021, accessed: 2021-12-07.

[37] Z. Lai *et al.*, "Vardict: a novel and versatile variant caller for next-generation sequencing in cancer research," *Nucleic Acids Res.*, vol. 44, pp. 1–11, 2016.

[38] Scitable by Nature Education, "Allele Frequency," https://www.nature.com/scitable/definition/allele-frequency-298/, 2022, accessed: 2022-03-31.

[39] Illumina, "Whole Exome," https://support.illumina.com/content/dam/illumina-support/help/Illumina_DRAGEN_Bio_IT_Platform_v3_7_1000000141465/Content/SW/Informatics/Dragen/TgtCountsWholeExome_fDG_dtSW.htm, 2020, accessed: 2022-03-22.

[40] Y. Yue *et al.*, "Evaluating metagenomics tools for genome binning with real metagenomic datasets and cami datasets," *BMC Bioinformatics*, vol. 334, 2020.

[41] A. R. Quinlan *et al.*, "BEDTools: a flexible suite of utilities for comparing genomic features," *Bioinformatics*, vol. 26, pp. 841–842, 2010.

[42] J. Zook, "Genome in a Bottle," https://github.com/ga4gh/benchmarking-tools/blob/master/resources/high-confidence-sets/giab.md, 2021, accessed: 2021-12-07.

[43] Pandas, "Pandas Documentation," https://pandas.pydata.org/docs/, 2022, accessed: 2022-03-31.

[44] Matplotlib, "Matplotlib 3.5.1 documentation," https://matplotlib.org/3.5.1/index.html, 2022, accessed: 2022-03-31.

[45] Seaborn, "Seaborn: Statistical Data Visualization," https://matplotlib.org/3.5.1/index.html, 2022, accessed: 2022-03-31.

[46] Scikit-Learn, "Scikit-Learn: Machine Learning in Python," https://scikit-learn.org/stable/, 2022, accessed: 2022-03-31.

[47] NCBI SRA, "Standard Scaling," https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html, 2022, accessed: 2022-04-08.

[48] Scikit Learn, "Choosing the right estimator," https://scikit-learn.org/stable/tutorial/machine_learning_map/index.html, 2021, accessed: 2021-12-08.

[49] ——, "Cross-Validation," https://scikit-learn.org/stable/modules/cross_validation.html, 2022, accessed: 2022-03-30.

[50] ——, "Confusion Matrix," https://scikit-learn.org/stable/modules/generated/sklearn.metrics.confusion_matrix.html, 2022, accessed: 2022-03-30.

[51] ——, "Precision-Recall," https://scikit-learn.org/stable/auto_examples/model_selection/plot_precision_recall.html, 2022, accessed: 2022-03-30.

[52] ——, "Roc-Auc-Score," https://scikit-learn.org/stable/modules/generated/sklearn.metrics.roc_auc_score.html, 2022, accessed: 2022-03-30.

[53] J.R. Quinlan *et al.*, "Induction of decision trees," *Machine Learning*, vol. 1, pp. 81–86., 1986.

[54] Scikit Learn, "Random Forest Classifer," https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html, 2022, accessed: 2022-03-30.

[55] P. Schober *et al.*, "Logistic regression in medical research," *Anesthesia and Analgesia*, vol. 132, pp. 365–366, 2021.

[56] W.J. Wilbur *et al.*, "Stochastic gradient descent and the prediction of mesh for pubmed records," *AMIA Annu Symp Proc.*, vol. 2014, pp. 1198–1207., 2014.

[57] Aniruddha, "AUC-ROC Curve in Machine Learning Clearly Explained," https://www.analyticsvidhya.com/blog/2020/06/auc-roc-curve-machine-learning/#:~:text=The%20Receiver%20Operator%20Characteristic%20(ROC,'%20from%20the%20'noise', 2022, accessed: 2022-03-27.

[58] I. Sarker, "Machine learning: Algorithms, real-world applications and research directions," *SN Computer Science*, vol. 160, 2021.

[59] A. Monaco *et al.*, "A primer on machine learning techniques for genomic applications," *Computational and Structural Biotechnology Journal*, vol. 19, pp. 4345–4359, 2021.

[60] K. Litkowski, "Feature ablation for preposition disambiguation," CL Research, Damascus, MD, Tech. Rep., 2016.

[61] L. Bomba *et al.*, "The impact of rare and low-frequency genetic variants in common disease," *Machine Learning*, vol. 77, 2017.

5 Appendix

Table 11: Feature Information

Feature Symbol	Symbol Name	Feature Type
ADJAF	Adjusted AF for indels due to local realignment	Numerical
AF	Allele Frequency	Numerical
ALT	Alternate non-reference alleles	Categorical
BIAS	Strand Bias Information	Categorical
DP	Total depth of coverage	Numerical
DUPRATE	Duplication Rate	Numerical
FORMAT	Extensible list of fields describing the sample	Categorical
HIAF	AF using only high quality bases	Numerical
HICNT	High quality variant reads	Numerical
HICOV	High quality total reads	Numerical
LongMSI	The somatic variant is flanked by long A/T (>=14)	Categorical
MQ	Mean mapping quality	Numerical
MSI	Number of Microsatellite Repeats	Numerical
MSI12	Variant in MSI region with 12 non-monomer MSI or 13 monomer MSI	Categorical
MSILEN	The unit length of MSI in bp	Numerical
NM	Mean mismatches in reads	Numerical
NM5.25	Mean mismatches in reads >= 5.25, thus likely false positive	Categorical
ODDRATIO	Strand bias odd ratio	Numerical
p8	Mean Position in Reads Less than 8	Categorical
PASS	Passed quality parameters	Categorical
PMEAN	Mean base position in the reads	Numerical
PSTD	Position STD in reads	Numerical
pSTD	Position in Reads has STD of 0	Categorical
Q10	Mean Mapping Quality Below 10	Categorical
QSTD	Quality score STD in reads	Numerical
QUAL	Mean quality score in reads	Numerical
REF	Reference bases	Categorical
REFBIAS	Reference depth by strand	Categorical
SBF	Strand Bias Fisher p-value	Numerical
SHIFT3	No. of bases to be shifted to 3' for deletions due to alt. alignment	Numerical
SN	Signal to Noise. The ratio of high quality bases/low quality bases	Numerical
SPANPAIR	No. of pairs supporting structural variants (SV)	Numerical
SPLITREAD	No. of split reads supporting structural variants (SV)	Numerical
SVLEN	The length of SV in bp	Numerical
SVTYPE	SV type: INV DUP DEL INS FUS	Categorical
VARBIAS	Variant depth by strand	Categorical
VD	Variant Depth	Numerical

```
from sklearn import tree
def build_decision_tree_classifier():
    return tree.DecisionTreeClassifier(
            max_depth=3,
            class_weight='balanced'
    )
```

Figure 21: Decision Tree Function

```
from sklearn.ensemble import RandomForestClassifier
def build_rf_classifier():
    return RandomForestClassifier(
        n_estimators=100,
        max_depth=3,
        random_state=0,
        n_jobs=16,
        class_weight='balanced'
    )
```

Figure 22: Random Forest Function

```
from sklearn.linear_model import LogisticRegression
def build_lr_classifier():
    return LogisticRegression(
        solver='liblinear',
        class_weight='balanced'
    )
```

Figure 23: Logistic Regression Function

```
from sklearn.linear_model import SGDClassifier
def build_sgd_classifier():
    return SGDClassifier(loss="log",
                         penalty="l2",
                         max_iter=1000,
                         tol=1e-5,
                         class_weight='balanced'
    )
```

Figure 24: SGD Classifier Function