UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escola Tècnica Superior d'Enginyeria Agronòmica i del Medi Natural

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

School of Agricultural Engineering and Environment

Evaluation and improvement of quality control methods for long read-defined transcriptomes

End of Degree Project

Bachelor's Degree in Biotechnology

AUTHOR: Estevan Morió, Eva

Tutor: Forment Millet, José Javier

External cotutor: CONESA CEGARRA, ANA

Experimental director: ARZALLUZ LUQUE, ANGELES

ACADEMIC YEAR: 2021/2022

BACHELOR THESIS

Biotechnology Bachelor's Degree

Evaluation and improvement of quality control methods for long read-defined transcriptomes.

Evaluación y mejora de métodos de control de calidad de transcriptomas elaborados a partir de lecturas largas.

Evaluación y mejora de métodos de control de calidad de transcriptomas elaborados a partir de lecturas largas.

València, July 2022

Academic year 2021/2022

AUTHOR: Eva Estevan Morió

ACADEMIC TUTOR: José Javier Forment Millet

EXTERNAL COTUTOR: Ana Conesa Cegarra

EXPERIMENTAL DIRECTOR: Angeles Arzalluz Luque

Abstract

High-throughput long-read transcriptome sequencing technologies have facilitated the discovery of novel transcripts. Nevertheless, these technologies have a much higher error rate than those based on short reads and therefore specific tools are required to characterise these novel variants and filter out false positives.

SQANTI3 (Structural and Quality Annotation of Novel Transcript Isoforms), a software for analysing long read-based transcriptomes, was born out of this need. SQANTI3 takes a transcriptome, together with genome annotation and, if available, other orthogonal data (expression, validation of the 3' and 5' ends, etc.) to return a characterised transcriptome. The tool also provides a wide set of descriptors of the isoforms and their splice junctions, which are further analysed in several diagnostic plots.

SQANTI3 includes an artificial intelligence-based classifier (MLfilter) that automatically discriminates transcripts that can be considered true isoforms from potential artifacts. This filter is based on a random forest algorithm, which brings multiple advantages to transcriptomic analysis, such as avoiding the use of manually set thresholds for each descriptor variable. However, like any machine learning model, it is a black box meaning that what happens between the input data and the predicted output is unknown.

By comparing different parameter set ups, we have characterised the MLfilter's performance for two transcriptome datasets and established guidelines to optimise the choice of training data according to the input data type. Specifically, we have evaluated the adequacy of the set of transcripts taken as true positives by the classifier, as well as the most relevant variables to obtain a good artifact-isoform classification. We have also detected avoidable biases such as overfitting. Ultimately, this work will help define best practices for the large community of researchers who use SQANTI3 to refine their transcriptomes and will allow further research to improve the MLfilter.

KEY WORDS: Transcriptomics, long-reads, isoforms

Resumen

Las tecnologías de secuenciación de alto rendimiento de transcriptomas mediante lecturas largas han facilitado el descubrimiento de nuevos transcritos. No obstante, dichas tecnologías tienen una tasa de error muy superior a las basadas en lecturas cortas, por lo que requieren herramientas que permitan caracterizar estas variantes noveles y filtrar las que son falsos positivos.

De esta necesidad nace SQANTI3 (Structural and Quality Annotation of Novel Transcript Isoforms), un software para el análisis de transcriptomas construidos a partir de lecturas largas. SQANTI3 toma un conjunto de datos de transcritos, junto con la anotación del genoma y, si están disponibles, otros datos ortogonales (expresión, validación de los extremos 3' y 5', etc.), para devolver un transcriptoma corregido. Asimismo, la herramienta proporciona un amplio conjunto de descriptores de las isoformas y sus sitios de splicing, que se analizan más a fondo en varias gráficas de diagnóstico.

SQANTI3 incorpora un clasificador basado en inteligencia artificial (MLfilter) que discrimina, de manera automatizada, los transcritos que pueden considerarse verdaderas isoformas de los potenciales artefactos. Dicho filtro se basa en un algoritmo de random forest, que aporta múltiples ventajas al análisis transcriptómico, entre ellas evita el uso de umbrales establecidos manualmente para cada variable descriptora. Sin embargo, como todo modelo de machine learning, se trata de una caja negra, es decir, se desconoce lo que pasa entre la entrada de datos y la salida de una predicción.

Comparando distintas combinaciones de parámetros de entrenamiento del MLfilter, hemos caracterizado su funcionamiento y hemos establecido una serie de guías para optimizar la definición de los datos de entrenamiento en función del tipo de datos de partida. Concretamente, se ha evaluado la adecuación del set de transcritos tomados como verdaderos positivos por el clasificador, así como las variables más relevantes para obtener una buena clasificación artefacto-isoforma. Además, hemos detectado errores evitables, como el 'overfitting' o sobreajuste. Todo esto, contribuirá a unas mejores prácticas por parte de la gran comunidad de usuarios que emplean SQANTI3 para refinar sus transcriptomas y abrirá la puerta a futuras investigaciones para mejorar el MLfilter.

PALABRAS CLAVE: Transcriptómica, lecturas largas, isoformas

Contents

List of Figures

List of Tables

Acronyms

APA alternative polyadenylation. 1–3

AS alternative splicing. 1–3

CAGE Cap Analysis of Gene Expression. 12

CCS Circular Consensus Sequencing. 4, 5

ENCODE Encyclopedia Of DNA Elements. 11, 12

FSM Full Splice Match. V, 8, 9, 12, 14–29, 31–33

ISM Incomplite Splice Match. V, 8, 9, 14, 18, 23, 27, 28

IsoSeq3 Scalable De Novo Isoform Discovery from Single-Molecule PacBio Reads. IV, V, 6, 10, 11, 14, 16, 17, 21, 22, 25–28

MLfilter SQANTI3's machine learning-based filter. V, VI, 8–10, 12–16, 18, 20, 26–28, 30, 33

mRNA messenger ribonucleic acid. IV, 1, 3–5, 19, 28, 29

NGS next-generation sequencing. 3, 30

NIC Novel In Catalog. V, 8, 9, 18, 23, 27, 28

NNC Novel Not in Catalog. V, 8, 9, 18, 23, 27, 28

ONT Oxford Nanopore Technologies. 4

PacBio Pacific Biosciences. 4–6, 10, 11, 14, 30

poly(A) polynucleotide chain composed entirely of adenosine monophosphates. 1, 4, 5, 7, 12

pre-mRNAs messenger ribonucleic acid precursors. 1, 5

RM Reference Match. V, 12, 14–21, 26, 27, 29, 31, 33

RNA-seq ribonucleic acid sequencing. IV, 3–5, 11, 30

RT reverse transcriptase. 5

SJ splice junctions. 1, 6, 7, 14, 19, 28, 30

SMRT Single Molecule Real Time. 4

SQANTI3 Structural and Quality Annotation of Novel Transcript Isoforms v3. IV–VI, 6–12, 14, 17, 18, 20–22, 25–27, 30–32

TALON Technology agnostic long read analysis pipeline for transcriptomes. IV, V, 6, 10, 11, 14, 26, 28, 29, 31, 32

TN True Negative. V, 7–9, 30

TP True Positives. IV, V, 7–9, 12, 14–27, 29–33

TSS Transcription Start Sites. 7, 12, 15, 18, 19, 23, 26, 28–30

TTS Transcription Termination Sites. 12, 15, 18, 19, 26, 28–30

Chapter I

Introduction

I.I mRNA maturation

The maturation of messenger ribonucleic acid precursors (pre-mRNAs) is a crucial step in transcription before they are exported from the nucleus as mature messenger ribonucleic acid (mRNA) in eukaryotic organisms (Figure I.I). This process involves the addition of the 5´ cap (Ramanathan et al. (2016)) and the polynucleotide chain composed entirely of adenosine monophosphates (poly(A)) tail at the 3' end (Kumar et al. (2019)), which play a key role in protecting the pre-mRNAs from enzymatic degradation and regulating the translation process. Furthermore, introns are removed by the spliceosome, a protein-RNA complex that recognizes splicing regulatory sequences (Newman (1998)). These sequences include donor (5´) and acceptor (3´) splice sites, which together are known as splice junctions (SJ).

Alternative splicing AS and alternative polyadenylation (APA) of pre-mRNAs enable the regulated generation of multiple mRNAs products or isoforms from a single gene, each of which may perform a different function (Figure I.II). AS and APA are major mechanism of transcriptome and proteome diversification in higher organisms (J. L. Chen (2009), Frankish et al. (2012), Mudge et al. (2011)).

AS occurs through different mechanisms, such as intron retention, exon skipping or extension (Frankish et al. (2012)) as illustrated in Figure I.II. These post-transcriptional modifications not only contribute to the establishment of the complexity of organisms (L. Chen et al. (2014)), but also play an important role in differentiation and speciation (Chapman et al. (2013), Nilsen and Graveley (2010), Park et al. (2020), Song et al. (2020), Teichroeb et al. (2016)). Besides, dysregulation of AS affects the onset and development of numerous diseases (Feng and Xie (2013), Rodríguez et al. (2016), Shkreta et al. (2013)) and therefore provides a target for potential therapies (Bergsma et al. (2018), Frankiw et al. (2019), P. Ren et al. (2021)).

On the other hand, polyadenylation consists in the endonucleolytic cleavage of pre-mRNAs and the addition of a series of adenosine monophosphates -the so-called poly(A) tail- at the cleavage site. APA produces several mRNAs isoforms by adding the poly(A) tail at different sites (Figure I.II). Consequently, those isoforms contain different 3' untranslated regions and coding sequences (F. Ren et al. (2020)). Although there is still much to be

studied about APA, it has been reported to be a crucial post-transcriptional regulation mechanism (Yeh and Yong (2016)) and to be related to disease progression and drug sensitivity (Zhang et al. (2021)). Furthermore, being a simpler mechanism than AS, it also represents a potential biomarker (Zhang et al. (2022)) and a potential therapeutic target (F. Ren et al. (2020), Y. Wang et al. (2022)).

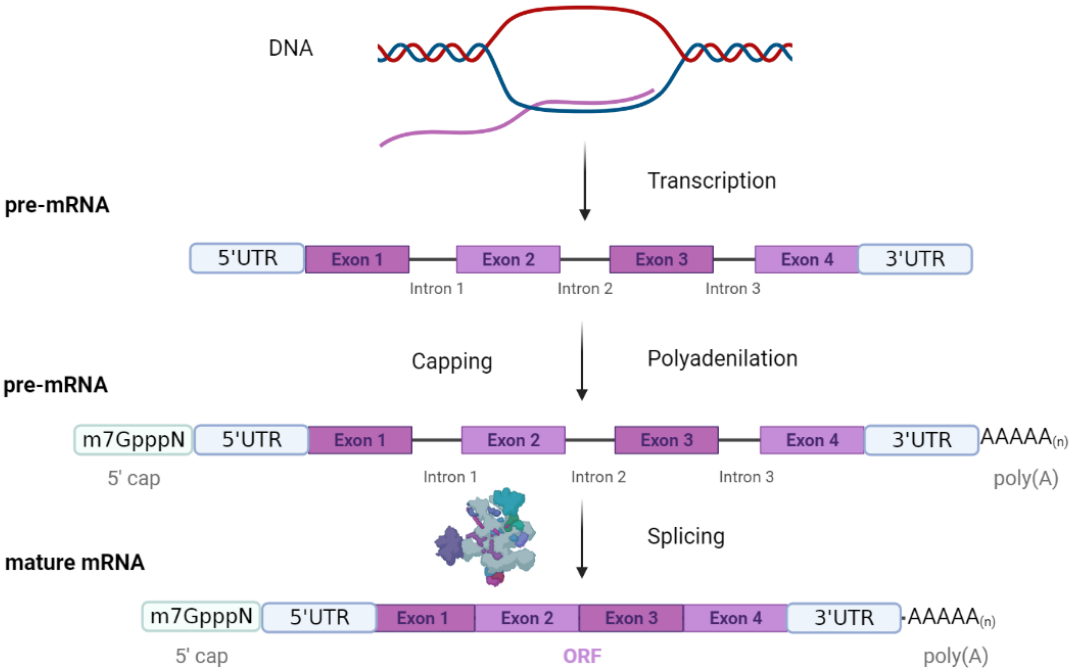


Figure I.I: Maturation process of pre-mRNA to mature mRNA. After DNA transcription to pre-mRNA, this molecule suffers from capping (addition of 5' cap), polyadenilation (addition of poly(A) tail at 3' end) and finally splicing by the spliceosome. After all these mechanisms a mature mRNA is obtained. Figure created with BioRender.com.

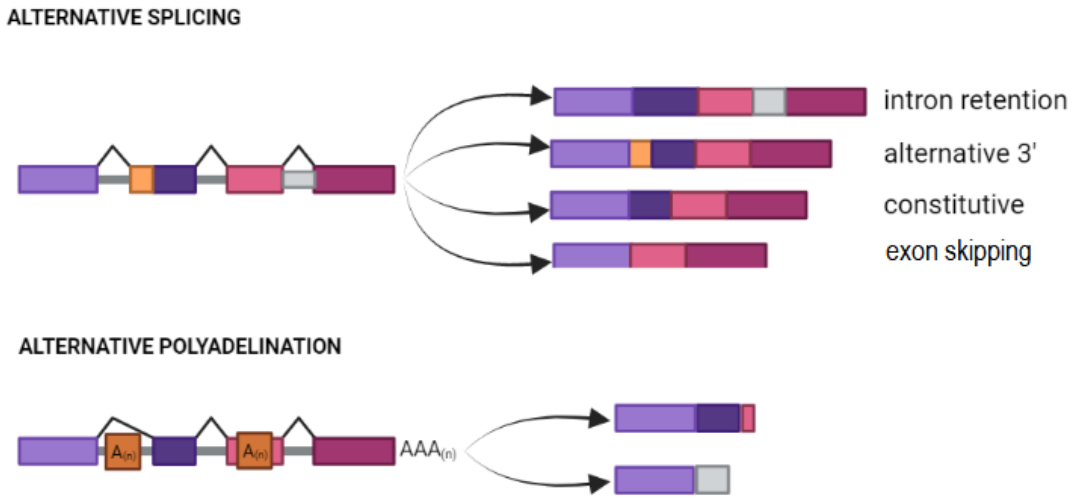


Figure I.II: Alternative splicing and alternative polyadenylation mechanisms and results. Alternative splicing mechanisms include intron retention, alternative 3' end or exon extension and exon skipping. Constitutive splicing is also shown. Polyadenylation consists in the addition of the poly(A) chain to a different site from the 3' end; two examples are illustrated. Figure created with BioRender.com.

I.II Novel transcript discovery

The transcriptome is the set of all transcripts or mRNA molecules produced in cells at a specific moment (Z. Wang et al. (2009)). It can be applied to the specific transcripts present in a single cell, a tissue or a whole organism. In addition, the transcriptome changes depending on external factors, different developmental stages or physiological conditions.

RNA sequencing (RNA-seq) is a useful technique to identify and quantify the RNA in a sample using next-generation sequencing (NGS) technologies (Z. Wang et al. (2009)). That is to say, RNA-seq is a technique to analyse the transcriptome "at a glance", i.e. without the need to target specific genes. Thus, the capacity to unravel the transcriptome is key to understanding gene expression and function more deeply. In addition, RNA-seq can identify post-transcriptional modifications such as APA or AS and the transcriptional structure of genes.

Transcriptome sequencing technologies such as short-read RNA-seq are useful for discovering post-transcriptional events (W. Chen et al. (2017)). Nevertheless, RNA-seq platforms -the most common among them being Illumina (Alamancos et al. (2014), Croucher et al. (2009))- produce reads of small length compared to mRNA transcripts and interrupt the sequences' continuity. This results in the inability to resolve assembly ambiguities at complicated loci and therefore hinder the characterization and quantification of alternative isoforms (Engström et al. (2013), Goodwin et al. (2016), Steijger et al. (2013)). Moreover, it is important to obtain full-length reads of transcripts to really

understand their functional diversity, as it relies in the whole structure, not only in a single post-transcriptional modification. For all the above, it is only possible to obtain a full mRNA sequence through alignment to annotated sequences or *de novo* transcriptome assembly approaches. Hence both approaches have major limitations for reconstructing the real expressed transcripts.

Recently, the development of long-read sequencing technologies has allowed the sequencing of entire molecules of full-length cDNA as single reads (Dijk et al. (2018)). These distinguishing features contrast with the short-read technologies, in which sequencing is paused after each base incorporation. Remarkably, most of the limitations of short-read transcriptome assemblers can be overcome with long-read technologies, as the length of the sequences does not limit them and the contiguity of the read is not lost (Leung et al. (2021)). High-throughput RNA-seq using long reads has facilitated the discovery of thousands of novel transcripts. Currently, Pacific Biosciences (PacBio) ((Eid et al. (2009),Sharon et al. (2013))) and Oxford Nanopore Technologies (ONT) (Oikonomopoulos et al. (2016)) are the prevailing long-read sequencing technologies.

ONT sequencing consists in a nanopore protein inserted into a synthetic lipid membrane through which a DNA or RNA molecule can pass. Each base provokes a different disruption in the membrane's electric field, hence the sequence is determined. In contrast with the PacBio technology, the standard ONT protocol does not allow sequencing of the same molecule multiple times, however, it is able to sequence longer molecules. (Oikonomopoulos et al. (2016))

The PacBio Single-Molecule, Real-Time sequencing technology is known as SMRT. SMRT sequencing is based on the fluorescence emitted by each labelled nucleotide that is incorporated during DNA synthesis by a polymerase molecule immobilised in the bottom of a microwell. The acces of the single molecule to the immobilised polymerase limits the length of the reads (Eid et al. (2009)). In addition, the development of novel library preparation methods by PacBio has reduced the typically high error rates of long-read technologies. First, full-length, double-stranded cDNAs are used to construct SMARTbell libraries. This is done by adding ligating hairpin adapters to both ends of the cDNA molecule, which allows its circularisation. Then, this circular molecule is repeatedly sequenced using circular consensus sequencing (CCS) (Travers et al. (2010)) to obtain a consensus read or read of insert (RoI). RoIs in which both cDNA primers and the poly(A) tail can be detected are considered Full-Length (FL) reads, while those that miss any of these are called non-Full-length (non-FL) reads.

Longer reads allow the sequencing of full-length mRNA molecules with single reads, hence avoiding the biases of short-read sequencing technologies. While the larger read sizes achieved by ONT have made it a leader in the field of genomics, PacBio's more precise CCS technology has been established as a powerful tool for RNA-seq, mainly because the small size of RNA molecules allows to read them sequentially in the same circular consensus molecule. For this reason, we have chosen to perform the present study with transcriptome data generated with the PacBio Sequel II system, which generates so-called highly accurate long reads (HiFi reads), given that they provide >99,9% accuracy and up to 8 million reads per sequencing run.

Nevertheless, the high raw error rate (11%), especially due to insertions and delections, is one of the most notable limitations of PacBio (Carneiro et al. (2012)). These errors are intrinsic to single-molecule sequencing, conversely to high-throughput short-read sequencing, where clusters of the same molecule are sequenced simultaneously, achieving very low error rates (Minoche et al. (2011)). However, the error rate has been estimated for each pass of the single molecule and, as explained above, the use of CCS technology has reduced the error rate. Hence, the exact error rate is highly dependent on the number of sequencing passes.

It is however essential to understand the sources of error in order to further minimise them in downstream analyses. Errors can be introduced during sample preparation, RNA extraction, library construction, sequencing and raw data processing (Shi et al. (2021)). Here, we will next explain the three mechanisms that have been most widely described to result in the identification of false isoforms -that is, hinder the identification of novel transcripts.

First, reverse transcriptase (RT) template switching is an intrinsic property of RTs that allows them to move the template positions without terminating DNA synthesis. It becomes troublesome when its activity is enhanced by secondary structures in RNA templates and generates gaps during cDNA synthesis. These gaps are then interpreted as splicing sites, leading to the generation of false alternative transcripts (Cocquet et al. (2006)).

A similar consequence is caused by off-priming of the oligo(dT) primer in adenine-rich regions of the mRNA template, known as intra-priming. In other words, shorter cDNA molecules are produced when the oligo(dT) primer anneals to a adenine-rich regions found in pre-mRNAs, that it is not the poly(A) tail. (Nam et al. (2002))

Finally, mRNA is an unstable molecule that tends to suffer degradation, which can be a source of errors during library preparation (Gallego Romero et al. (2014)). RNA transcript decay is a highly regulated physiological process that occurs during the cell cycle and contributes to regulating other cellular mechanisms (Garneau et al. (2007)). Nevertheless, this degradation process also occurs during the extraction of mRNA, mainly due to RNase present in the cellular extract or in the environment, but is undesired. The degradation process begins with the removal of the 3' tail of poly(A), followed by the decapping of the 5' cap structure and the exonucleolytic degradation from 5' 3' or degradation from 3' to 5'. It must be noted that transcripts with missing 3' end will be discarded during cDNA synthesis, as it is done with an oligo(dT) primer, whereas transcripts with shorter 5' ends due to degradation will be sequenced.

I.III Long-read transcriptome reconstruction software

Given the mechanisms explained above, which generate a diversity of transcripts that is not naturally present in the biological sample, errors during RNA-seq present a major obstacle to generating accurate transcriptomic data. Indeed, the potential of long-read technologies to generate transcriptomes has raised the need to develop software to efficiently identify isoforms based on these reads.

Numerous long-read transcriptome reconstruction pipelines have been developed, most notably IsoSeq3 and TALON. IsoSeq3 is a pipeline developed by PacBio to generate transcriptomes using data obtained with this platform ("PacificBiosciences/IsoSeq" (2022)). This pipeline generates transcripts by clustering and polishing PacBio HiFi reads and, given that it does not use a reference genome, it is designed to identify novel isoforms. Conversely, TALON is a platform-independent assembler that relies on reference transcripts for identifying isoforms (Wyman et al. (2020)). The TALON pipeline is designed to annotate reads as known or novel transcripts and quantify them simultaneously. First, error correction is performed by mapping and comparing the raw reads to the reference genome. After that, the tool analyses quality control information to identify and characterise new transcript models and finally filter and quantify all of them.

Additionally, downstream quality control and transcript annotation analyses are required due to the characteristics of the reads obtained using PacBio. Several software tools have been developed to cover this demand, one of the most widely used among them being Structural and Quality Annotation of Novel Transcript Isoforms v3 (SQANTI3) (Tardaguila et al. (2018)) .

I.IV SQANTI3 for quality control of long read-defined transcriptomes

SQANTI (Tardaguila et al. (2018)) was born out of the need to refine and characterise long read-defined transcriptomes and constitutes the first module of the Functional IsoTranscriptomics (FIT) pipeline, including IsoAnnot and tappAS (de la Fuente et al. (2020)). The SQANTI3 pipeline provides an in-depth characterisation and curation of long-read transcriptomes, meaning that it performs isoform classification and quality control, but it also includes a specific module to filter out potential artifacts, i.e. false positive isoforms generated during library preparation and sequencing.

SQANTI3 takes as input a long read-defined transcriptome, together with the corresponding genome sequence and reference transcriptome annotation (and, if available, other orthogonal data, such as isoform expression, validation of the 3' and 5' ends, etc.) to return a characterised transcriptome. The tool also provides a wide set of transcript-level attributes and descriptors for each isoform and their SJs, further analysed in several diagnostic plots. Thus, the SQANTI3 output information allows users to understand their isoform models' properties and identify potential errors. Here, we have classified the most relevant variables for this work depending on their characteristics (Table I.I).

On the one hand, it classifies the isoforms into pre-defined categories (Table I.II), (Figure I.III.a) and subcategories (Figure I.III.b-d) .

I.IV.a SQANTI3 machine learning filter

The term machine learning refers to a branch of artificial intelligence based on constructing a mathematical model that can be trained with existing data to make predictions on any dataset of the same type (Tarca et al. (2007)). These can be: 1) supervised models, which accurately predict the classification of new elements based on the features of available data,

Table I.I: SQANTI3 classified transcript-level variables

CLASSIFICATION	NAME	MEANING
Short-read coverage related variables	min_sample_cov	Number of short-read replicates mapping the SJ least covered by short-reads.
	sd_cov	Standard deviation of coverage between SJs. SJ coverage uniformity.
	min_cov	Value of coverage of the least covered SJ of the transcript.
	ratio_TSS	Coverage ratio between first 100pb upstream and downstream TSS.
Expression-related variables	gene_exp	Expresion level of the original gen.
	Iso_exp	Short-read expression for this isoform.
	Ratio_exp	Ratio of iso_exp to gene_exp.
	FL	Full-length read count associated to the isoform.
End and start related variables	polyA_motif	Top ranking polyA motif found up to 50 bp upstream of the transcript end.
	polyA_dist	Location of the last base of the hexamer. Position 0 is the putative poly(A) site. This distance is hence always negative because it is upstream.
	diff_to_TSS	Distance of query isoform 5' start to reference transcript start site. Negative value means query starts downstream of reference.
	diff_to_TTS	Distance of query isoform 3' end to reference annotated end site. Negative value means query ends upstream of reference.
	diff_to_gene_TSS	Distance of query isoform 5' start to the closest start site of any transcripts of the matching gene, by looking at all annotated starts of a gene.
	diff_to_gene_TTS	Distance of query isoform 3' end to the closest end of any transcripts of the matching gene.
Structure related variables	Bite	TRUE if the transcript contains at least one novel SJ for which a novel intron overlaps an annotated exon.
	Exons	Number of exons.
	Coding	Coding potential capacity according to GeneMarkS-T (Tang et al. (2015)).
	Length	Isoform length
	Predicted_NMD	TRUE if there's a predicted ORF and CDS ends at least 50bp before the last SJ; FALSE otherwise.

and 2) unsupervised models, in which there is no predefined classification and elements under study are clustered together to discover their natural grouping. Examples of both supervised and unsupervised machine learning models applied to biological problems can be found in Tarca et al. (Tarca et al. (2007)).

In this case, we have labelled targets: artifacts or real isoforms, defined as such by SQANTI3; therefore, a supervised machine learning model is the best option (Chicco (2017)). The chosen model relies on a random forest method (Breiman (2001)), a supervised model that is built from a large number of decision trees that work as an ensemble. Decision trees can be understood as a flowchart where each internal node represents a test on a variable, each branch constitutes a test result and each terminal node represents a class tag. Therefore, training random forests involves testing random decision trees in which different importance is given to each attribute with a set of True Positives (TP) and True Negative (TN) data provided. Through this process, if accuracy degrades when excluding a certain variable, its importance value is incremented and ultimately the most accurate model is

Table I.II: SQANTI3 category description

CATEGORY	MEANING
Full Splice Match (FSM)	The reference and query isoform have the same number of exons and all internal junctions agree.
Incomplite Splice Match (ISM)	The query isoform has fewer 5' or 3' exons than the reference, but all common internal junctions agree.
Novel In Catalog (NIC)	The query isoform does not have a FSM or ISM match, but is using a combination of known donor/acceptor sites.
Novel Not in Catalog (NNC)	The query isoform does not have a FSM or ISM match, and has at least one donor or acceptor site that is not annotated.
Antisense	The query isoform does not have overlap a same-strand reference gene but is anti-sense to an annotated gene.
Genic Intron	The query isoform is completely contained within an annotated intron.
Genic Genomic	The query isoform overlaps with introns and exons.
Intergenic	The query isoform is in the intergenic region.

used to classify a provided dataset.

The SQANTI3 machine learning-based filter (MLfilter) is a random forest algorithm that, starting from the classification file generated by SQANTI3, a TP and a TN set of 3000 transcripts, filters the transcripts considered as isoforms and discards some as artifacts (Figure I.IV). This algorithm learns from the properties of the transcripts in the TP and TN sets regarding the multiple variables (Table I.I) considered to filter the transcripts. According to their relevance in the classification model, these variables are given an importance value. Transcripts that have potential to be artifacts should be used as TN set, whereas transcripts that are most likely to be true should be used as TP set. Choosing the model training data is crucial for an optimum performance, therefore it is important to study how they are defined, which is the main objective of this work.

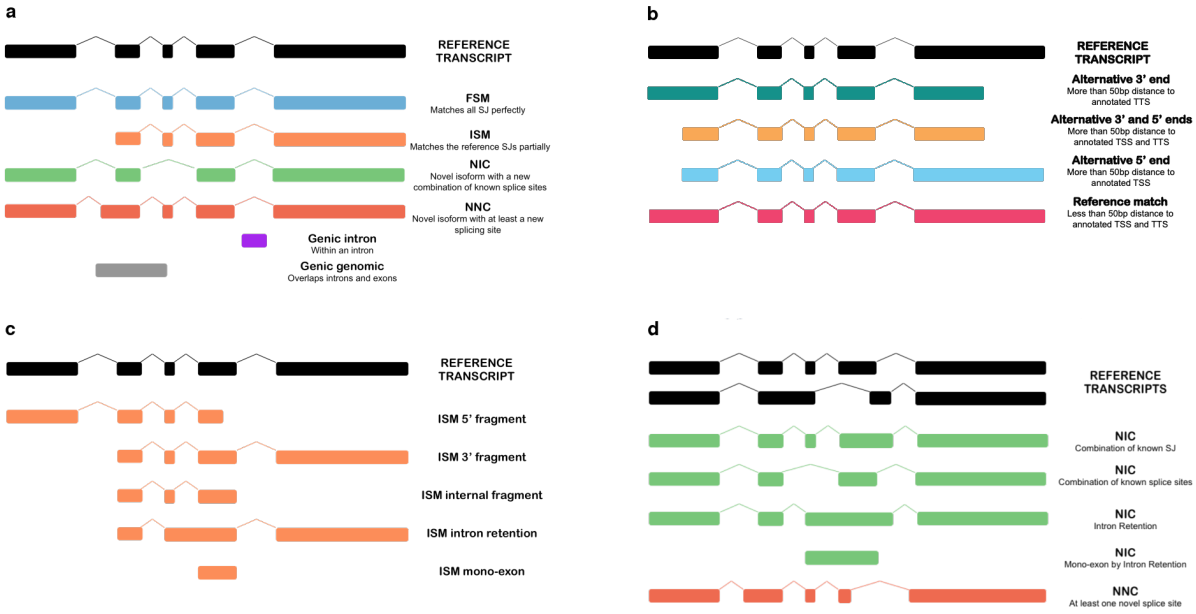


Figure I.III: Representation of SQANTI3 categories and subcategories compared to a reference transcript, adapted from "SQANTI3" (2022). **a.** SQANTI3 main categories. **b.** Full splice match (FSM) subcategories. **c.** Incomplete splice match (ISM) subcategories. **d.** Novel in catalog (NIC) and novel not in catalog (NNC) subcategories

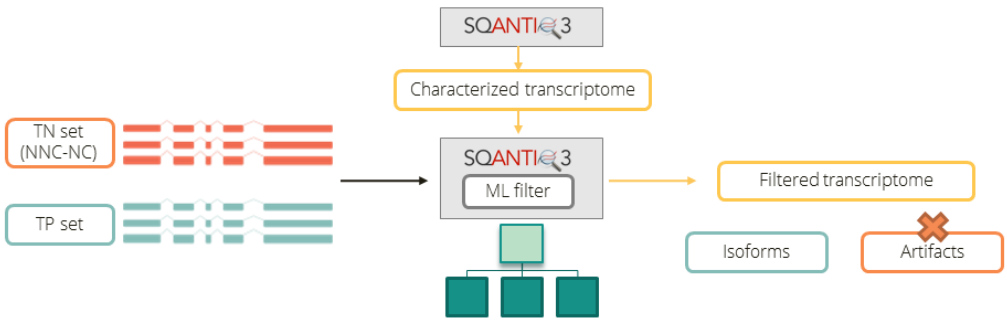


Figure I.IV: SQANTI3 machine learning filter (MLfilter) workflow. After running SQANTI3, the resulting characterized transcriptome is filtered. The MLfilter is trained with a true negative (TN) set of transcripts (novel not in catalog with non-canonical junctions) and a True Positives (TP) set of transcripts. The final output is a classification of the transcripts as artifacts or as isoforms.

Chapter II

Objectives

The SQANTI3 MLfilter brings multiple advantages to transcriptomic analysis, however it is necessary to study its performance and the optimum chose of training data. In this work, we intend to define best practices for researchers who use SQANTI3 to characterise transcriptomes and open the door to further studies to improve the SQANTI3 pipeline.

Accordingly, two main objectives of this thesis are: 1) understand how the MLfilter output varies depending on the parameters established for the model's training and 2) establish a list of recommendations for the use of the SQANTI3 MLfilter, in order to optimise the filtering of artifacts in long read-defined transcriptomes. These recommendations will be based on two different long-read transcriptomics datasets, but they can be extrapolated to long-read transcriptomes processed with the same or similar pipelines: one that is reference-independent and specific for PacBio reads (IsoSeq3) and one that is reference-based and sequencing platform-independent (TALON). With this in mind, we have performed a comparative analysis after running the MLfilter with different parameter set ups, in an attempt to draw conclusions as to which of these configurations is the most suitable for the study of each dataset.

Chapter III

Methods

III.I Long-read RNA-seq data availability

Long-read RNA-seq data generated using the Iso-Seq library preparation method from PacBio and sequenced using the Sequel II platform was retrieved from the Encyclopedia Of DNA Elements (ENCODE) project database (Moore et al. (2020)).

The first dataset comes from C2C12, an immortalized mouse myoblast cell line, and its RNA-seq data was obtained from ENCODE accession ENCSR221XGR. The second dataset comes from GM12878, an human B cell derived cell line, and its RNA-seq data was downloaded from ENCODE accession ENCSR838WFC.

III.II Long-read data pre-processing

C2C12 long-read data was processed using TALON (Wyman et al. (2020)), using default parameters. The full TALON pipeline is available on GitHub through the ENCODE4 Data Coordinating Center (DCC) at https://github.com/ENCODE-DCC/long-read-rna-pipeline ("ENCODE Long read RNA-seq pipeline" (2022)) and at https://github.com/mortazavilab/TALON ("TALON" (2022)).

GM12878 data was processed using the IsoSeq3 software and default parameters. The full IsoSeq3 pipeline is available on GitHub at https://github.com/PacificBiosciences/IsoSeq ("PacificBiosciences/IsoSeq" (2022)). In addition, we mapped these fasta reads to the genome reference and then run cDNA Cupcake (Tseng (2022)) to collapse these transcripts models and remove redundancy, using default parameters and `[-dun-merge-5-shorter]`, to avoid collapsing shorter 5' transcripts.

III.III Running SQANTI3

In order to characterise and evaluate the quality of both long read-generated transcriptomes, the quality control module of the SQANTI3 toolkit (SQANTI3 QC, v5.0) was used. The SQANTI3 software can be found at https://github.com/ConesaLab/SQANTI3 ("SQANTI3" (2022)).

III.III.a C2C12 data

For C2C12 data, we run SQANTI3 using the mouse reference genome (GRCm39) and the Gencode vM27 transcriptome. Moreover, other sources of supporting data were supplied to SQANTI3 to perform the quality control. First, in accordance with previous studies stating that human and mouse poly(A) motif sequences show high levels of conservation (Tian et al. (2005), R. Wang et al. (2018)), a ranked list of common human poly(A) motif sequences (included in SQANTI3) was used to curate Transcription Termination Sites (TTS) for our isoforms. In addition, mouse Cap Analysis of Gene Expression (CAGE) peak data was obtained from the FANTOM5 database and used to validate Transcription Start Sites (TSS) across the transcriptome (Forrest et al. (2014)). Finally, one Illumina short-read dataset (ENCODE accession: ENCSR000AHY) was supplied to account for short-read coverage and isoform/gene expression information during quality control.

III.III.b GM12878 data

For GM12878 data quality control, we run SQANTI3 using the human reference genome (GRCh38.p13) and the Gencode v37 transcriptome. Similarly to what was described above for C2C12, several sources of supporting data were used to expand and refine transcriptome quality control, i.e. human CAGE peak data from the FANTOM5 database (Forrest et al. (2014)), the list of common poly(A) motif sequences included in SQANTI3 (see section above) and an Illumina short-read dataset including two replicates (ENCODE accession: ENCSR000AEH).

III.IV Running SQANTI3's machine learning-based filter

In this work, the MLfilter (SQANTI3 ML filter, v5.0) was run using different sets of TP transcripts and excluding/including several variables during random forest model training via the `-remove_columns` or `-r` flag in the SQANTI3 filter script. Of note, the remaining parameters were left as default.

The exact combinations of TP set and excluded variables used to apply the MLfilter are shown in Table III.I. On the one hand, a random sample of 3000 Reference Match (RM) or Full-splice Match (FSM) isoforms was used as TP set. In turn, two MLfilter runs were performed using each of these TP sets, i.e. including and not including the SQANTI3 attributes related to the genomic distance to the start/end of the associated reference transcript and to the associated gene. These attributes reflect the similarity between the long read-defined isoform's TSS/TTS and those included in the reference, and are hereafter referred to as distance variables.

III.V Generation of diagnostic plots for the comparative analysis

The SQANTI3 filter module outputs a classification table including transcripts as rows and all SQANTI3 descriptors, together with filter result variables (i.e. random forest

Table III.I: Parameter combinations for running the MLfilter

TRUE POSITIVE SET	DISTANCE PARAMETERS	ABBREVIATION
Reference Match	Included	`RM_dist`
Reference Match	Not included	`RM`
Full Splice Match	Included	`FSM_dist`
Full Splice Match	Not included	`FSM`

classifier probabilities and final filter result), as columns. Using the data output by the different MLfilter runs, we generated a group of diagnostic plots that will be discussed in the Results section. Those plots were generated using R programming language (R Core Team (2020)) and the set of packages included in the 'tidyverse' (Wickham et al. (2019)). A sample of the scripts used to generate the figures reported in this thesis is included in the Appendix section A.

Chapter IV

Results and discussion

IV.I General description of the analysis

As explained before, the aim of this study is to establish a list of recommendations to get the most out of the machine learning-based filter (MLfilter) of SQANTI3. These recommendations are based on two different long-read transcriptomics datasets, but they can be extrapolated to long-read transcriptomes with similar properties. For this purpose, the particularities of each dataset are hereby described and a comparative analysis performed after running the MLfilter with different parameters (see Methods section II.IV, III.I), in an attempt to draw conclusions as to which of these set ups is the most suitable for the study of a dataset with similar characteristics.

Both transriptomics datasets were generated with the PacBio sequencing technology and then processed with two different pipelines: IsoSeq3 and cDNA Cupcake for the GM12878 cell line dataset and TALON for data from the C2C12 cell line (see Methods, section II.I). Then, we run SQANTI3 and its machine learning filter with four different parameter set ups (see Methods section II.IV, table III.I).

In past SQANTI versions, the pipeline did not include orthogonal data -such as CAGE peak and polyA motif/peak data- to validate the transcripts' ends, therefore, the MLfilter algorithm was not applied to FSM and ISM categories. Both of these categories include isoforms that share all their junctions with a reference transcript, and therefore mainly present variability at their 3' and 5' ends. Hence, without information related to the transcripts' ends, FSM and ISM transcripts could not be filtered. As it is now possible to include this type of information, it is indeed possible to filter these SQANTI categories. The variability in the ends of FSM and ISM transcripts also makes it difficult to know whether they are novel isoforms or come from errors during library preparation, such as 5' end RNA degradation or intra-priming, or during raw data processing, e.g. the inability to correct long read sequencing errors. With this in mind, we have tested two aspects of MLfilter set up: first, how to create the true positive (TP) set; and second, how the selection of variables for classifier training influences the filter output. Ultimately, we intend to discern which set up leads to a better performance.

To define the TP set, we have used transcripts that match the reference exactly (reference match subcategory, RM) and transcripts that match only their SJs (full-splice match, FSM)

as seen in Figure I.III.b. On the one hand, RM is a FSM subcategory and transcripts included in it are the most likely ones to be true, as they are fully consistent with the reference. Nevertheless, we cannot know if training the algorithm with this type of transcript results in discarding novel isoforms, since they do not resemble the RMs. On the other hand, FSMs include other subcategories that are shorter in one or both of the ends (Figure I.III.b). Hence, training the algorithm with a more diverse set of transcripts may allow the detection of novel isoforms.

However, when using only RMs as TP, and not the rest of FSM subcategories, an overfitting effect is likely to occur. In machine learning, overfitting can be defined as the optimization of the learning algorithm such that it perfectly fits the training data, but its performance decays on other similar datasets (Chicco (2017)). It is sometimes explained because, instead of learning from the dataset provided, the algorithm "memorises" its characteristics and loses the ability to be flexible when other data is supplied. In the case of this study, when training the algorithm with RM as TP, the filter is learning from transcripts that have a minimum distance to the beginning and end of the gene and of the associated reference isoform. In consequence, we anticipate the filter to be very demanding when it comes to determining whether a transcript is an isoform or not, as it will do it mainly in terms of reference transcriptome similarity. Therefore, by including variables related to TSS/TTS distance in the filter and using a TP set of transcripts with minimum values of these variables, we anticipate this overfitting effect to be produced. Overfitting would lead to a more demanding filtering proccess, in which the contribution of other variables would be minimized. For this reason, in order to study the overfitting effect, we have also tested the inclusion or exclusion of variables related to TSS/TTS distance from classifier training, hereby called distance variables.

We therefore run the machine learning filter with four different parameter set ups, which we will refer to with the following nomenclature:

- `RM_dist`: RM as TP set and including `diff_to_gene_TTS` and `diff_to_gene_TSS` columns.
- `RM`: RM as TP set and excluding `diff_to_gene_TTS` and `diff_to_gene_TSS` columns.
- `FSM_dist`: FSM as TP set and including `diff_to_gene_TTS` and `diff_to_gene_TSS` columns.
- `FSM`: FSM as TP set and excluding `diff_to_gene_TTS` and `diff_to_gene_TSS` columns.

Ultimately, we have performed a comparative analysis of the output obtained from running the filter with these parameters. All of these by comparing the values of the variables for each transcript (Table I.I) used by the MLfilter and the filtering output, that is, the classification of the transcripts as isoforms or artifacts (Figure I.IV). Using this data, we generated a group of diagnostic plots in an effort to evaluate the different outputs obtained.

IV.II GM12878 human cell line transcriptome proccessed with IsoSeq3

First of all, it is interesting to have an insight of how the FSM TP set is composed, as this category has five different subcategories, therefore the training will be different depending on the type of transcripts classified as FSM in the transcriptome under study. For the GM12878 dataset, we have seen that FSM transcripts are distributed in all subcategories, with the alternative 3' end subcategory accumulating 50% of total FSM transcripts (Figure IV.I.a). This could influence the filtering, because, when training from a set of transcripts that are mostly shorter at the 3' end, the algorithm is expected to be more permissive with transcripts of this type than with the rest. However, as this is a representative sample of the real transcripts in the data, this should not be a problem -it would only introduce a bias if these alternative 3' end transcripts predominate due to an error in library preparation or raw data processing.

Regarding the relevance of the different variables in the MLfilter, when all four different filtering set ups used were compared, we observed that some distance variables (`diff_to_gene_TTS` and `diff_to_gene_TSS`) are given larger values of importance in those cases in which these variables are included (Figure IV.I.b). In general, this effect leads to reducing the importance of some variables in the classification, especially the most highly contributing ones.

The first comparative analysis of the data consists in seeing the number of artifacts detected by each filter set up, the number they share and those unique to each one (Figure IV.I.c). Using the `RM_dist` configuration, 18455 transcripts are uniquely discarded, meaning that the rest of filtering set ups do not detect them as artifacts. This suggests that, for the data processed with IsoSeq3, using a TP of RM and providing distance parameters for random forest classifier training is more stringent than the rest of parameter combinations used. It is noteworthy that, while the other filtering sets up discard few unique transcripts, `RM_dist` discards a large number of them. Therefore, we have focused on investigating this phenomenon.

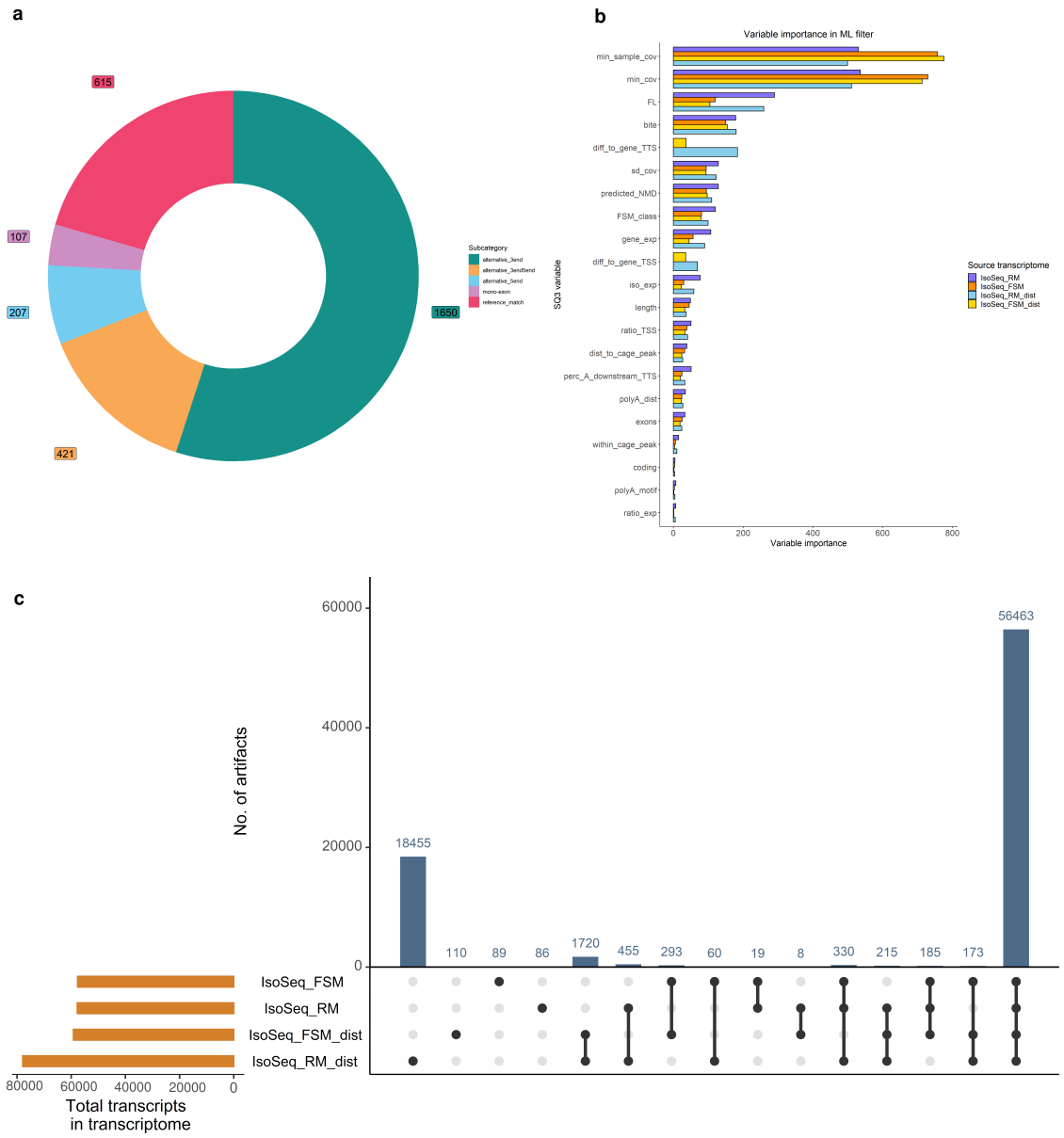


Figure IV.I: Diagnostic plots for GM12878 cell line transcriptome proccessed with IsoSeq3 and cDNA Cupcake comparing the four set ups of the machine learning filter of SQANTI3: IsoSeq_RM (TP=RM, not including the distance variables) IsoSeq_RM_dist (TP=RM, including the distance variables), IsoSeq_FSM (TP=FSM, not including the distance variables), IsoSeq_FSM_dist (TP=FSM, including the distance variables). **a.** Percentage of subcategories of the transcripts included in the TP set generated with FSM transcripts for each filter set up. **b.** Variable importance values of each filter set up used in the machine learning filter of SQANTI3. **c.** UpSet plot showing the number of unique and shared artifacts between the different filter set ups used in the machine learning filter of SQANTI3.

To shed some light on the reason why the `RM_dist` set up generates such a high number of unique artifacts, we will discuss the upset plot in Figure IV.I.c in more detail. First, `RM_dist` and `FSM_dist` share 1720 unique artifacts that no other filtering set up discards. In fact, `RM_dist` and `FSM_dist` are the pair of filter set ups that share the highest amount

of artifacts. As inclusion of distance variables is the only parameter they share, this may indicate that distance parameters seem to be determinant for the filtering. In contrast, 19300 transcripts are discarded by `RM_dist` and not by `FSM_dist`, meaning that, regardless of the high amount of shared artifacts, there is still a great difference between them. This is why it would be interesting to compare `RM_dist` and `FSM_dist` outputs and learn from those unique and shared artifacts and their properties. In particular, we will 1) study the similarities of the shared discarded transcripts to understand the effect of the inclusion of distance variables and 2) assess the properties of the unique artifacts of each filtering set up to understand the influence of the choice of TP set.

IV.II.a `RM_dist` vs `FSM_dist`: TP set effect

Comparing both filter set ups that include the distance parameters (`FSM_dist` and `RM_dist`) enables us to understand how the selection of the TP set and the inclusion of the distance variables influences the outcome of the MLfilter.

Regarding the total number of artifacts in each SQANTI3 category(Figure IV.II.a), FSM, ISM, NIC and NNC show a higher percentage of unique artifacts. This was expected, as FSM and ISM categories are the most likely to be influenced by the distance to the TSS/TTS of the transcript. That is because they are characterised by being shorter or longer at the TSS/TTS, thus including related parameters or not is expected to affect their classification as isoforms. For this reason, we will focus on FSM, ISM, NNC and NIC categories to provide a clearer visualisation of the data. In addition, reference match is the only subcategory in which there is a higher percentage of unique artifacts in the `FSM_dist` filtering than in `RM_dist` (Figure IV.II.b). That is because, when using a TP set of reference match classified transcripts, there should be less artifacts from this subcategory. However, the total number of RM artifacts is small compared with the rest of subcategories. Remarkably, there is a high percentage of unique artifacts from the `RM_dist` filter set up in the alternative 3' end and the alternative 3' and 5' end subcategories (Figure IV.II.b), which may be due the high percentage of alternative 3' end transcripts in the FSM TP set.

Regarding variable importance in the random forest classifier, we observed that `RM_dist` gives more importance to `diff_to_gene_TTS` and `diff_to_gene_TSS` than to other variables that have more importance in `FSM_dist` (Figure IV.II.a). As discussed in the section above, this may be due to overfitting: the algorithm is being trained with transcripts with a minimum distance to the end and start of the associated reference gene and transcript, so it will be very demanding regarding this variable. In addition, this effect leads to ignoring variables that are relevant in other set ups, such as `min_sample_cov` or `min_cov` (Figure IV.I).

To verify whether these observations can be attributed to an overfitting effect, we have examined the distribution of values of the distance variables to see whether there is a difference between unique and common artifacts. `RM_dist`'s unique artifacts have greater `diff_to_gene_TTS` values than `FSM_dist` ones and even greater than the common artifacts in some cases (Figure IV.II.c). Notably, for `RM_dist`, `diff_to_gene_TTS` is a far more important variable. Meanwhile, `FSM_dist` is not focusing only in this attribute and may be discarding those artifacts because of other variables, regardless of the little distance to the

3' end of the gene. In contrast, there is not such a difference in `diff_to_gene_TSS`, and both filtering set ups behave similarly regarding this variable (Figure IV.II.d).

To discuss this issue better, we shall provide some context. During RNA extraction, the 5' end of the mRNA molecules is commonly degraded. Many of these fragments are thus sequenced and can be wrongly detected as novel isoforms, hence it is required to be able to know which of them are caused by degradation. Transcripts of this origin will be part of the FSM set, as a result, training the algorithm with these transcripts as true isoforms will lead to a permissive filtering of transcripts generated by degradation in the 5' (`diff_to_gene_TSS` > 0). Nevertheless, we cannot know the exact values of distance to the TSS corresponding to a transcript generated by degradation problems. What can be assumed is that filtering with RM as TP set will not tolerate this degradation-generated transcripts. Therefore, we can conclude that those artifacts with higher distances to the TSS may have this origin, while those artifacts with little distances filtered by the `FSM_dist` set up may have been generated by other causes.

On the other hand, 3' end degradation is not detected because the resulting chain lacks on poly(A) tail. 3' end assortments are most commonly generated by intra-priming, for which the transcript is truncated due to the detection of a adenine-rich fragment by the oligo(T) primer. The distance to the TTS of this transcripts cannot be known unless the adenine-rich fragment is detected. What can be assumed by observing Figure IV.II.f is that `RM_dist` considers transcripts with higher distances to the TTS (which may or may not be originated by intra-priming) to be artifacts; however, those unique artifacts are more similar to the common ones. Other remark that can be made from Figure IV.II.c is that `FSM_dist` is discarding transcripts with distances to the 3' end close to 0, which suggests that they may be considered artifacts for other reasons.

Next, we have analysed the performance of the artifacts with regard to the values of the variables related to short-read data (Table I.I). These attributes are indicators of the short-read coverage, i.e. they are an external proof of validation of the isoform, either in the SJs or in the TTS (`ratio_TTS`). Given their high importance values (Figure IV.I.b), we will discuss two short-read variables: min_cov and min_sample_cov. As can be seen in Figure IV.II.e and Figure IV.II.f, unique artifacts with lower min_cov and min_sample_cov are found in `FSM_dist`, while `RM_dist` discards transcripts with greater short-read coverage values than the common ones. In order to understand why these transcripts with strong coverage support are being considered as artifacts, these were plotted together with the distance variables. As illustrated in Figure IV.III.a and Figure IV.III.b, for greater values of min_sample_cov, there are greater values of `diff_to_gene_TTS` and `diff_to_gene_TSS`. This follows the expected behaviour, especially for `RM_dist` filtering, whereby the filter discards transcripts with good coverage values only if they have also high distances. In addition, as discussed above, `diff_to_gene_TTS` is high for `RM_dist` in all cases, while for the `FSM_dist` set up the values of the distance to the TTS are always low (Figure IV.III.a). While the difference is not noteworthy for `diff_to_gene_TSS` , its values are larger for `RM_dist` compared to `FSM_dist` unique artifacts (Figure IV.III.b). This suggests that `RM_dist` is filtering based on the distance to the gene TTS value, while `FSM_dist` must be filtering due to other attributes.

Regarding the rest of variables, no differences were found except in the case of gene_exp.

This variable refers to the level of gene expression and high values of gene_exp are considered to be evidence of the existence of the isoform. As it has been seen in the case of other short-read related variables, gene_exp is higher for `RM_dist` unique artifacts than for common and unique `FSM_dist` ones (Figure IV.III.c). Nonetheless, if the transcript is generated by degradation or intra-priming, it would show the expression levels and short-read coverage of a true isoform. For this reason, they should be discarded even if they have high values of these variables.

Considering all the above, a set of recommendations to optimally run SQANTI3's MLfilter can be formulated:

- Overfitting is not desired, consequently using FSM as TP set is highly recommended when using the distance variables for random forest classifier training.
- For a more stringent filtering, but with more variability in the training set, FSM should be used as TP. This option is recommended if the user has a reliable short-read dataset, as it will mainly learn from short-read coverage.
- For a more lenient filtering that learns from a wider range of variables it is recommended to use RM as TP set and not include the distance variables.

The next question to be answered is whether it is better to use distance variables when the TP set is generated with FSM or not. For this purpose, a different set of filtering set ups was next compared: filtering with a TP set componed of FSM transcripts with (`FSM_dist`) and without the distance variables (`FSM`).

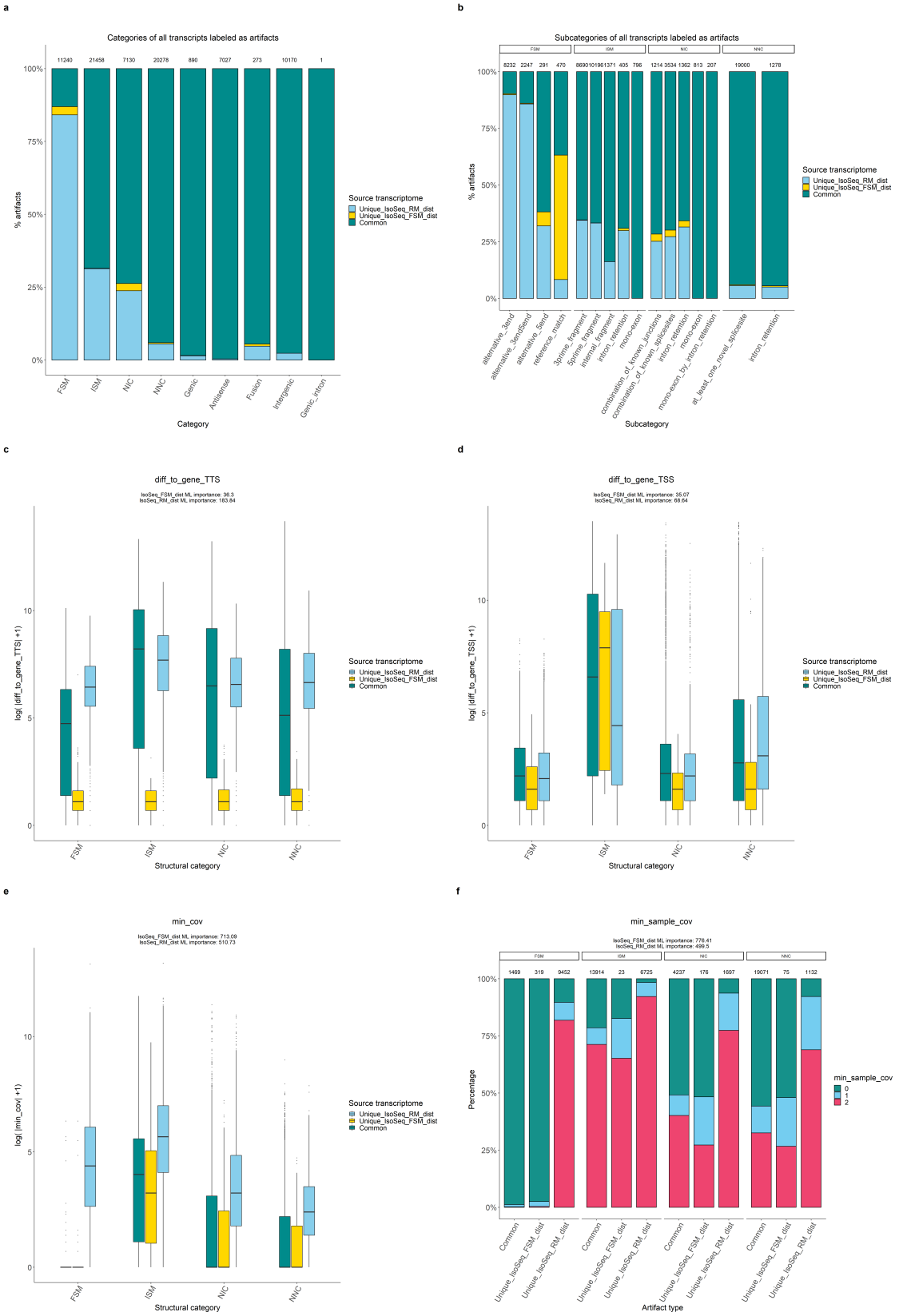


Figure IV.II: Diagnostic plots for GM12878 transcriptome processed with IsoSeq3, comparing two filtering set ups of the machine learning filter of SQANTI3: `RM_dist` (TP=RM, including the distance variables), `FSM_dist` (TP=FSM, including the distance variables). **a.** Percentage of common and unique transcripts labeled as artifacts for each category. **b.** Percentage of common and unique transcripts labeled as artifacts for each subcategory. **c.** Values of log(|`diff_to_gene_TTS`|+1) for artifacts of each category. **d.** Values of log(|`diff_to_gene_TSS`|+1) for artifacts of each category. **e.** Values of log(|`min_cov`|+1) for artifacts of each category. **f.** Percentage of common and unique transcripts labeled as artifacts for each category and value of `min_sample_cov`.

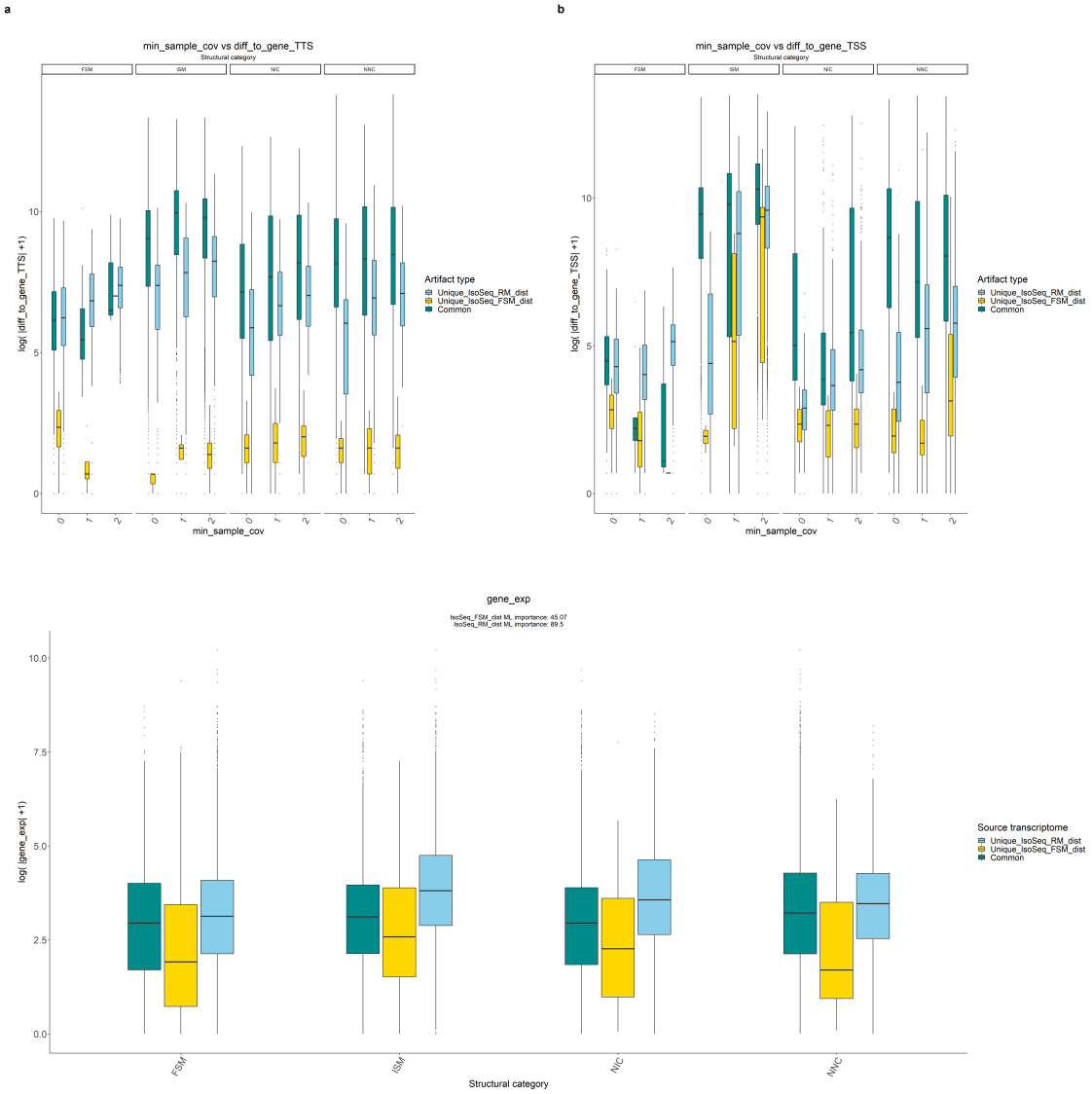


Figure IV.III: Diagnostic plots for GM12878 transcriptome processed with IsoSeq3, comparing two filtering set ups of the machine learning filter of SQANTI3: `RM_dist` (TP=RM, including the distance variables), `FSM_dist` (TP=FSM, including the distance variables).**a.** Values of log(|`diff_to_gene_TTS`|+1) for artifacts of each category and value of `min_sample_cov`. **b.** Values of log(|`diff_to_gene_TSS`|+1) for artifacts of each category and value of `min_sample_cov`.. **c.** Values of log(|`gene_exp`|+1) for artifacts of each category.

IV.II.b `FSM_dist` vs `FSM`: distance variables effect

In order to have a deeper understanding of how the distance variables influence the output, we compared both filterings which TP set was generated with FSM transcripts: `FSM` (TP=FSM, not including the distance variables and `FSM_dist` (TP=FSM, including the distance variables).

As seen in Figure IV.I.b, there is not as much difference in variable importance between `FSM_dist` and `FSM` as in the previous comparison. Nevertheless, in `FSM_dist`, the distance variables have high importance values, downplaying the importance of the rest of variables. Remarkably, `FSM_dist` seems to be slightly more stringent, as it discards 2053 transcripts that `FSM` filter set up do not discards.

Regarding the percentage of artifacts in each category and subcategory, the same categories as previously shown appear in the spotlight (Figure IV.IV.a), hence we will focus again on FSM, ISM, NIC and NNC classified transcripts. In this case, all of FSM subcategories have greater percentages of unique `FSM_dist` artifacts than unique `FSM` ones (Figure IV.IV.b), in contrast to what we have observed in the previous comparison, in which the reference match subcategory was different from the rest in this regard. Ultimately, which may be noted is that these two set ups are more alike than the previous ones, hence the inclusion of the distance variables might be less critical than the choice of TP set.

To study the origin of the discrepancies between the two set ups, we have analysed the differences of each attribute between unique and common artifacts for both set ups. First, regarding the distance variables, both unique artifacts have lower values of `diff_to_gene_TSS` than the common ones (Figure IV.IV.c), whereas `FSM_dist` unique artifacts have higher `diff_to_gene_TSS` than `FSM` ones (Figure IV.IV.d). This reveals the same pattern as in the previous comparison: the most stringent set up (`FSM_dist` in this case) has unique artifacts with higher `diff_to_gene_TTS` than the common ones, while the most lenient set up (`FSM` in this case) has unique artifacts with lower `diff_to_gene_TTS` than the common ones. As hypothesised before, this may be a consequence of the most lenient set up taking the rest of variables into account. Thus, taking into account that `FSM` set up does not include the distance variables in the filtering process, this effect was to be expected.

As discussed above, the set of transcripts classified as FSM is mainly composed by alternative 3' end transcripts, that is, those shorter than the reference in the 3'end. Accordingly, the machine learning algorithm is being trained to recognise this type of transcripts as the true ones: that is why it is expected to have artifacts with low values of `diff_to_gene_TSS`, as alternative 3' end transcripts have small distances (or no distance at all) to the 5' end (TSS) due to degradation. Nevertheless, it was not expected that transcripts with high `diff_to_gene_TTS` (distance to the 3'end of the gene) would be considered artifacts, as alternative 3' end transcripts - which have been used as true isoforms to train the model - have high `diff_to_gene_TTS` values. Therefore, the logical output is the one accomplished with the `FSM` set up, while `FSM_dist` appears to be performing abnormally in this regard, possibly due to overfitting. In this case, it does not seem to be as troublesome as in the `RM_dist` set up, but it should always be avoided.

Moreover, in the same way as before, we have studied the differences with respect to the short-read related variables. In Figures IV.IV.e-f, it is illustrated that `FSM_dist`

discards transcripts with stronger short-read support (min_cov and min_sample_cov) than the `FSM` filter set up, thus confirming that those unique `FSM_dist` artifacts are being considered artifacts due to distance variables, regardless external proof such as min_cov and min_sample_cov values.

By all accounts, overfitting might be produced due to the high percentage of alternative 3' end FSM subcategories, since these represent more than 50% of the FSM transcripts. As mentioned before, transcripts classified as alternative 3' end have high `diff_to_gene_TTS` values. As a result, including these values could lead to over-training the classifier to only consider isoforms with these exact characteristics. As overfitting is not desired, it would be better not to include the `diff_to_gene_TTS` variable in random forest classifier training.

All things considered, a set of recommendations can be formulated:

- The `FSM_dist` filtering set up is more demanding than `FSM`, but not excessively. Therefore, the decision to include the distance variables when using FSM as TP set should be based on how permissive the user wants the filter to be.
- It is important to know the subcategory distribution of FSM transcripts of the dataset before applying the machine learning filter. If the kind of overfitting described here is expected to occur, it would be better to remove the columns of the problematic variables.

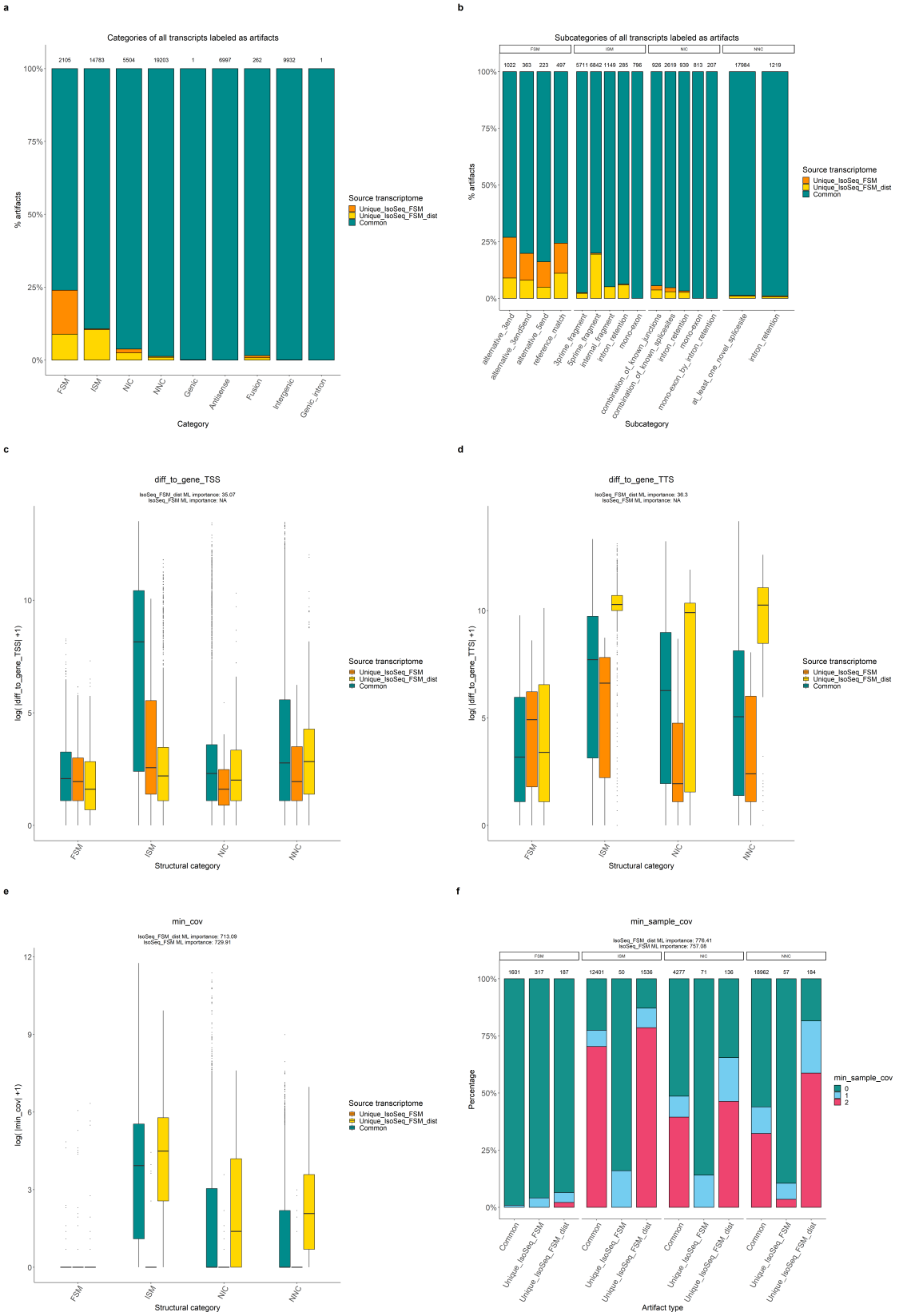


Figure IV.IV: Diagnostic plots for GM12878 transcriptome processed with IsoSeq3, comparing two filtering set ups of the machine learning filter of SQANTI3: `FSM` (TP=FSM, not including the distance variables), `FSM_dist` (TP=FSM, including the distance variables). **a.** Percentage of common and unique transcripts labeled as artifacts for each category. **b.** Percentage of common and unique transcripts labeled as artifacts for each subcategory. **c.** Values of log(|`diff_to_gene_TTS`|+1) for artifacts of each category. **d.** Values of log(|`diff_to_gene_TSS`|+1) for artifacts of each category. **e.** Values of log(|`min_cov`|+1) for artifacts of each category. **f.** Percentage of common and unique transcripts labeled as artifacts for each category and value of `min_sample_cov`.

IV.III C2C12 mouse cell line data processed with TALON

Aiming to verify the findings of the previous section, the same filter set ups were studied for a different dataset: a C2C12 cell line transcriptome processed with the TALON pipeline.

As discussed above, we have first examined the composition of the FSM TP sets (Figure IV.V.a). In this case, FSM transcripts mainly belonged to the RM subcategory (75%), followed by the mono-exon subcategory (almost 25%). The rest of subcategories contain only 20 transcripts out of the 3000 included in the TP set. As transcripts in the RM subcategory and in the FSM category will be mainly alike, it is assumed that using either set of transcripts as the TP set will not make a major difference. Hence, we will hereby focus on the effect of including the distance variables for the random forest classifier training.

Regarding the variable importance values, there is a much more equal distribution than in the previous dataset studied (Figure IV.V.b). This means that, in this case, the contribution of the variables to the learning process of the algorithm is more evenly distributed. As observed before, `min_sample_cov` has the highest importance, however, there are other variables, such as `FSM_class` and `gene_exp`, that have a larger weight. However, it is again observed that distance variables (`diff_to_gene_TSS` and `diff_to_gene_TTS`) are given great values of importance in those cases in which these variables were included, downplaying the importance of other variables (Figure IV.V.b). Therefore, the effect of the inclusion of the distance variables in the filtering process will be similar to the one observed for GM12878 data processed with IsoSeq3.

Another behaviour that differs from the previously studied case is that no filter set up is distinct in the number of unique artifacts (Figure IV.V.c). Nevertheless, the pair of `RM_dist` and `FSM_dist` shares 62358 unique artifacts that the rest of the filter set ups do not discard. This was expected, as 75% of FSM transcripts are RMs, hence what it is interesting is that training the model with distance variables leads to a more stringent filtering.

As well as before, an overfitting effect is expected when using data that is too similar to the reference genome in terms of distances to the TSS/TTS and also including that type of information in the training of the classifier. For that reason, we have studied the behaviour of the MLfilter of SQANTI3 when including those variables, and compared it to when they are not included.

IV.III.a Distance variable effect

Assuming that RM and FSM transcripts are equivalent, we have choosen those filter set ups that use FSM as TP set to study the effect of the distance variables. Comparing `FSM_dist` and `FSM` set ups allows us to understand how including the distance variables in the filtering process influences the classification of transcripts as isoforms/artifacts in this type of dataset.

First of all, as discussed in the previous section, `FSM_dist` is more stringent than `FSM`. Particularly, the `FSM_dist` set up discards 19300 unique transcripts, while the `FSM` set up

only discards 596 unique ones and the rest (58671) are shared between both of them. This reaffirms the demanding nature of the `FSM_dist` filtering. In the discussion below, we will consider the values of the different variables for the unique artifacts of each filtering set up, in order to understand which attributes are being used by the MLfilter in each case.

As observed in Figure IV.V.b, the distance variables take high values of importance when they are used to train the model. This diminishes the importance of the other variables, hence including or excluding the distance variables will make a difference in MLfilter performance.

Regarding the total number of artifacts in each SQANTI3 category, FSM, ISM, NIC and NNC show a higher percentage of unique artifacts (for `FSM_dist` set up) (Figure IV.IV.a), as seen in the IsoSeq-processed data (Figure IV.VI.a). For the same reasons, we will from now on show the data for these four SQANTI3 categories. Furthermore, in regards to the percentages of unique artifacts for the subcategories, we observed a similar pattern as in the first comparison (IsoSeq3 `FSM_dist` vs `RM_dist`) of IsoSeq3 processed data (Figure IV.IV.b). In this case, as shown in Figure IV.VI.b, the `FSM_dist` set up has more unique artifacts in each subcategory unless in reference match. Remarkably, this effect cannot be due to the choice of RM transcripts as TP set, since `RM_dist` discarded more unique transcripts than `FSM_dist` in IsoSeq3 processed data (Figure IV.III.b). Conversely, it must be because of the exclusion of the distance variables. Hence, not providing the distance variables for MLfilter training leads to a more stringent filtering of reference match transcripts.

As a discordant note, while the `min_cov` variable is the second most important one for the previous dataset, in this case it is not even taken into consideration for the filtering (Figure IV.V.b). Moreover, the second most important variable in both `FSM` and `FSM_dist` filter set ups is `FSM_class` (Figure IV.V.b). Primarily, this is a variable that was not expected to contribute to the classification of transcripts as artifacts. `FSM_class` is a transcript attribute in regard to the gene it comes from and it has three classes:

- **A**: the original gene does not have any other transcript.
- **B**: the original gene has other transcripts, none of them classified as FSM.
- **C**: the original gene has other transcripts and at least one is classified as FSM.

In spite of being of great importance in this case, it does not show any differences in most categories (Figure IV.VI.c). The great percentage of C class artifacts is noteworthy, since it is difficult to understand why the classifier is filtering out a large number of transcripts that come from a gene with other isoforms, and at least one FSM. However, when looking at the values of `FSM_class` for transcripts classified as isoforms, we observe that they also show high percentages of C class transcripts. Hence, this predominance of C class transcripts is not specific to the artifacts, but to the entire transcriptome. Due to all this, the `FSM_class` variable seems to highly contribute to the training of the MLfilter, conversely, it does not seem to be linked to any other characteristic that distinguishes artifacts from true long read-defined isoforms, this one being an example of how it is often hard to unravel the inner functioning of the algorithms of machine learning.

Similarly to what was observed for the data processed with IsoSeq3 (Figure IV.I.b), the `min_sample_cov` variable is the most important one when applying the MLfilter to the TALON C2C12 transcriptome (Figure IV.V.b). Unique artifacts with higher `min_sample_cov` values than the common artifacts values are found in `FSM_dist`, whereas the `FSM` set up discards transcripts with lower short-read coverage values than the common ones (ISM) or more similar to the common ones (FSM, NNC, NIC) (Figure IV.VI.d). This first filter set up, which is more stringent, discards transcripts with stronger proof of short-read coverage, but not with low values, as opposed to `FSM` filter set up.

In order to confirm whether the unique artifacts are filtered out because of the values of distance variables, we plotted `min_sample_cov` together with `diff_to_gene_TSS` (Figure IV.VI.e) and `diff_to_gene_TTS` (Figure IV.VI.f). First, it is observed that `FSM` unique artifacts have lower values of the distance variables than the common artifacts. These low values could be evidence to classify those transcripts as isoforms, although they are considered artifacts. However, the number of `FSM` unique artifacts is too small, and this set up also discards many artifacts with potentially bad values of distance variables (the common artifacts). For the `FSM_dist` unique artifacts, we have examined if they are discarded due to the anomalous distance values. These values are indeed high, with similar values to the common artifacts ones. Therefore, the unique `FSM_dist` transcripts seem to be discarded because of the distance variables. In parallel, when not including the distance variables for training the model (`FSM` filter set up), those transcripts with high distances to the TSS/TTS are not discarded because of the short-read coverage proof (high values of `min_sample_cov`). All considered, it is difficult to know whether those are real isoforms; however, they may be considered artifacts due to overfitting.

Additionally, it is important to take into account the distance to the TSS of the gene, since, as explained before, a larger distance may be due to degradation of the 5' end of the mRNA. In Figure IV.VI.e we observe how, when including the distance variables (`FSM_dist` set up), transcripts with greater distance are discarded in comparison to those discarded by both filters, especially in the ISM and novel categories (NNC and NIC). This can be explained by the fact that degradation of mRNAs occurs at the TSS, hence, there will be greater variability of distances to the TSS and more transcripts with high distance values will be discarded. This contrasts with what is observed in the same comparison for the distance to the gene TTS (Figure IV.VI.f), where only unique `FSM_dist` artifacts classified as NNC have higher values of `diff_to_gene_TTS` than the common ones, while the opposite is true for the rest of SQANTI categories. NNC transcripts are those least likely to be true isoforms, as their SJs are completely novel, i.e. not present in the reference transcriptome. This initially led us to think that NNC transcripts may have longer distances to the ends of the gene, and would therefore be classified as artifacts. In order to unravel this opposite behaviour, we have studied the possible relationship between the distance to the TSS and to the TTS. Nevertheless, no relationship has been found (data not shown), hence the hypothesis has been discarded. In addtion, the idea that transcripts with short distances to the TSS are considered artifacts because they have high distances to the TTS is therefore dismissed.

On the other hand, the `ratio_TSS` variable is apparently a good parameter to determine whether a transcript is a real isoform, because if the `ratio_TSS`>1, it means that there is

evidence of short-reads supporting the 5' end. While the 3' end is usually more reliable, the 5' end of mRNA is more susceptible to degradation, hence, evidence of short-reads at that end is revealing information for filtering. However, it is not relevant when filtering in any category (Figure IV.V.b).

As seen throughout the study, distance variables that provide evidence from the reference, such as `diff_to_gene_TSS` and `diff_to_gene_TTS` do have substantial importance for the filtering. Meanwhile, `ratio_TSS` is not given importance in the random forest classifier, although it represents empirical evidence from the sample itself. Therefore, we have compared `ratio_TSS` to `diff_to_gene_TSS` and `diff_to_gene_TTS` to discover whether there is any relationship between them, however, nothing significant was found (data not shown).

Finally, we provide further context on the pipeline used for processing. The transcriptome processing tool (TALON) works in such a way that the transcripts are extended to match the reference data. This makes the use of RM as TP with information on gene and transcript start and end distances somewhat redundant, which, as we have seen, leads to discarding many more transcripts than the other filtering set ups, as well as to potential overfitting. Moreover, as we start from a transcriptome where FSM transcripts have a high percentage of RM (75%), there is not much difference between the two sets of true positives used. For this reason, in the case of TALON transcriptomes, we do not recommended to include information of the distance to the TSS/TTS when using FSMs as TP set.

Considering all of the above, we conclude this section by providing a set of recommendations for transcriptome datasets similar to those processed with TALON:

- For this dataset, it is preferable to exclude the classification columns: `diff_to_gene_TSS` and `diff_to_gene_TTS`, as they represent redundant information.
- For a more stringent filtering, `diff_to_gene_TSS` and `diff_to_gene_TTS` columns can be included, but with the potential risk of overfitting.
- For filtering and in this particular case the TP set used is not crucial, as the two candidates sets: FSM or RM are highly alike (75% of FSM are RM). Therefore, it is crucial to be aware of the composition of your dataset's FSM transcripts in order to decide which TP set to use.

IV.IV Final remarks of the discussion

This thesis has focused on the great opportunity that artificial intelligence brings to biotechnology and, more specifically, its role in long-read transcriptome analysis applications. Moreover, it is clear evidence of how bioinformatics tools and programming languages such as R facilitate processing of high-throughput data such as transcriptome sequencing. Likewise, their implementation for processing biological data of any kind is crucial. Finally, bioinformatics is key to integrating the increasingly large amount of available sequencing information, thereby achieving efficient research and reliable results.

Long-read sequencing technologies, such as PacBio, are powerful for sequencing high-throughput transcriptomes, although they present numerous challenges. In this context, software tools such as SQANTI3 increase the precision of these RNA-seq datasets, while implementing artificial intelligence algorithms in such pipelines has the power to improve its performance. However, it has become clear that it is challenging to fully understand the inner workings of a machine learning algorithm, which is inherent to artificial intelligence. Therefore, it is essential to design a proper model and choose the most suitable training data in order to achieve a true-to-life result and ultimately optimise NGS technologies. This work has intended to fill this gap by focusing on the SQANTI3 MLfilter and shed some light on how the training parameters influence the final output.

Firstly, one of the most common complications of machine learning models, overfitting, has been addressed in the present thesis. It has been demonstrated that it is essential to test the model to detect this type of anomaly. One of the study's major limitations is that the model has been not validated with sequenced data. Validation could be accomplished in further research by studying how the MLfilter works on control RNA sets, such as Lexogen's Spike-In RNA Variants (SIRVs) (Lexogen (2017)). SIRVs are synthetic RNA molecules that mimic the main aspects of transcriptome complexity, including alternative splicing and isoform expression. By adding small amounts of these controls to RNA samples before library construction, SIRVs could be used in downstream analysis as 'ground truth'. In supervised machine learning models, 'ground truth' refers to the reality to be modelled by the algorithm and in this case, the SIRVs could be used to measure the accuracy with which the MLfilter outputs a transcriptome that resembles the 'ground truth'.

The first version of the SQANTI MLfilter (Tardaguila et al. (2018)) included short-read information to validate potentially false SJs, mainly found in NIC and NNC transcripts. This algorithm proved that including this type of supporting data was enough to discard false novel transcripts, therefore using NNC non-canonical as the default TN set was a choice based on that evidence. For this reason, only the TP set choice was tested in the current study. Nevertheless, providing a TN set with more variety, i.e. including other potential artifacts (such as ISM without TSS/TTS support), could be considered an improvement and could be studied in further research.

We would also like to underline that this current thesis is part of the SQANTI3 project, for which the paper is now under preparation. This work therefore constitutes a preliminary study that has opened the door to understanding and validating the SQANTI3 MLfilter. Currently, and thanks to the insights obtained during this study, further work is being done by the research group to validate the random forest method, including testing with a more complex variety of TP and TN sets and excluding or including other variables. A strategy that is also being developed by the team is the implementation of a simulator (jmestret (2022)) to generate as realistic long reads as possible from known transcripts, thus allowing the identification of false positives transcripts generated by the pre-processing pipelines. Therefore, the MLfilter performance could be evaluated by examining whether the classifier discards those false positive transcripts. In this case, reliable simulated transcripts are to be considered 'ground truth', which would eliminate the need to include SIRVs in the RNA sample.

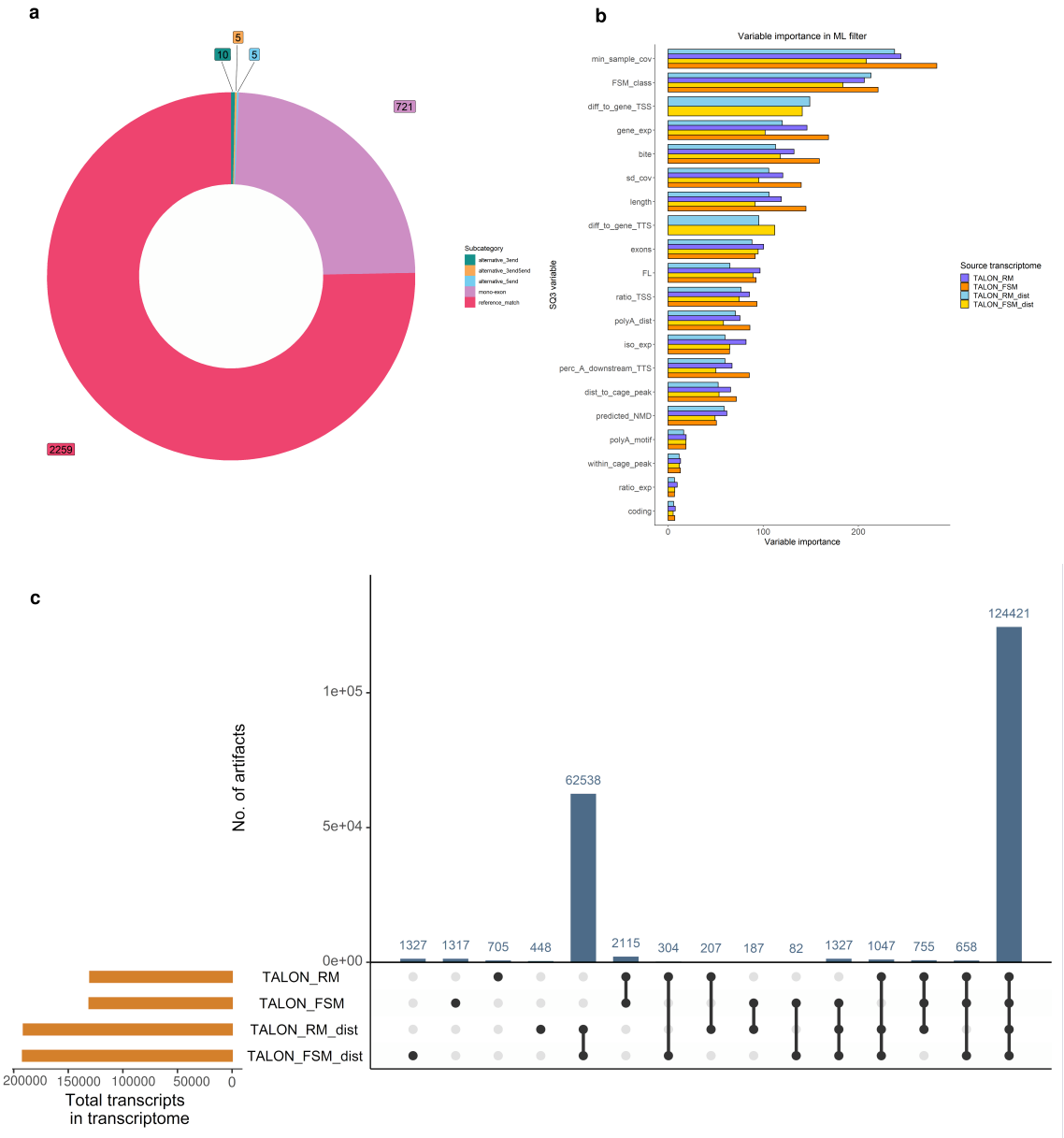


Figure IV.V: Diagnostic plots for the C2C12 cell line transcriptome proccessed with TALON comparing the four set ups of the machine learning filter of SQANTI3: TALON_RM (TP=RM, not including the distance variables) TALON_RM_dist (TP=RM, including the distance variables), TALON_FSM (TP=FSM, not including the distance variables), TALON_FSM_dist (TP=FSM, including the distance variables) **a.** Percentage of subcategories of the transcripts included in the TP set generated with FSM transcripts. **b.** Variable importance values of each filtering set up used in the machine learning filter of SQANTI3. **c.** Number of unique and shared between the different filtering set up used in the machine learning filter of SQANTI3.

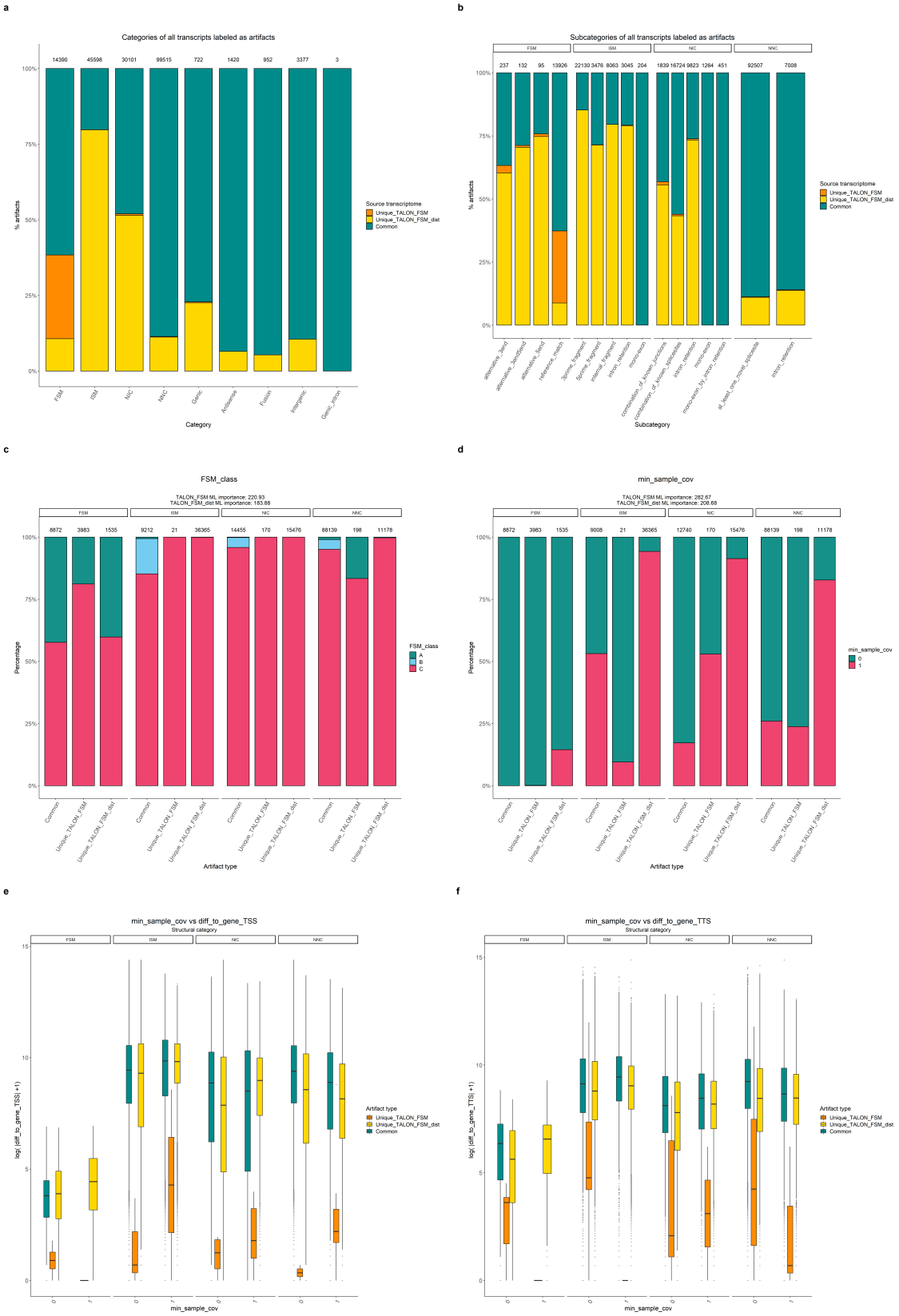


Figure IV.VI: Diagnostic plots for C2C12 transcriptome processed with TALON data, comparing two filtering set up of the machine learning filter of SQANTI3: `FSM_dist` (TP=FSM, including the distance variables), `FSM` (TP=FSM, not including the distance variables). **a.** Percentage of common and unique transcripts labeled as artifacts for each category. **b.** Percentage of common and unique transcripts labeled as artifacts for each subcategory. **c.** Values of log(|`diff_to_gene_TTS`|+1) for artifacts of each category. **d.** Values of log(|`diff_to_gene_TSS`|+1) for artifacts of each category. **e.** Values of log(|`min_cov`|+1) for artifacts of each category. **f.** Percentage of common and unique transcripts labeled as artifacts for each category and value of `min_sample_cov`.

Chapter V

Conclusions

In summary, two clear objectives were achieved: 1) to assess how the performance changes according to the parameters established for the model's training and 2) to provide basic guidelines for using the model based on the available data. In the absence of validation, guidelines have been proposed based on cautiousness and in the pursuit of a balance between losing valuable information and trusting false data. Overfitting tends to discard reads that may be true at the expense of being stringent with the data provided for training. Since the ML filter is part of a data refinement and quality control tool, it is preferred not to discard truthful data. Otherwise, the filter would avoid detecting novelty. That is why it is intended to avoid issues like overfitting.

Taking everything into account, the overall conclusions drawn from the present work are:

- Overfitting is not desired and it has been observed when combining a TP set generated with RM transcripts and the inclusion of distance variables for the MLfilter training. Therefore, this parameter combination should be avoided.
- It is crucial to know the subcategory distribution of FSM transcripts of the dataset before applying the MLfilter: if a high percentage of FSM are RM, distance variables should be excluded for the MLfilter training.
- The decision to include the distance variables when using FSM as TP set should also be based on how permissive the user wants the filter to be: including them for a more stringent filtering.

Bibliography

Alamancos, G. P.; Agirre, E., & Eyras, E. (2014). Methods to study splicing from high-throughput RNA sequencing data. In K. J. Hertel (Ed.), *Spliceosomal pre-mRNA splicing: Methods and protocols* (pp. 357–397). Humana Press. https://doi.org/10.1007/978-1-62703-980-2_26

Bergsma, A. J.; van der Wal, E.; Broeders, M.; van der Ploeg, A. T., & Pim Pijnappel, W. W. M. (2018, January 1). Chapter three - alternative splicing in genetic diseases: Improved diagnosis and novel treatment options. In F. Loos (Ed.), *International review of cell and molecular biology* (pp. 85–141). Academic Press. https://doi.org/10.1016/bs.ircmb.2017.07.008

Breiman, L. (2001). Random forests. *Machine Learning*, *45*(1), 5–32. https://doi.org/10.1023/A:1010933404324

Carneiro, M. O.; Russ, C.; Ross, M. G.; Gabriel, S. B.; Nusbaum, C., & DePristo, M. A. (2012). Pacific biosciences sequencing technology for genotyping and variation discovery in human data. *BMC Genomics*, *13*, 375. https://doi.org/10.1186/1471-2164-13-375

Chapman, M. A.; Hiscock, S. J., & Filatov, D. A. (2013). Genomic divergence during speciation driven by adaptation to altitude. *Molecular Biology and Evolution*, *30*(12), 2553–2567. https://doi.org/10.1093/molbev/mst168

Chen, J. L., Mo; Manley. (2009). Mechanisms of alternative splicing regulation: Insights from molecular and genomics approaches. *Nature reviews. Molecular cell biology*, *10*(11), 741–754. https://doi.org/10.1038/nrm2777

Chen, L.; Bush, S. J.; Tovar-Corona, J. M.; Castillo-Morales, A., & Urrutia, A. O. (2014). Correcting for differential transcript coverage reveals a strong relationship between alternative splicing and organism complexity. *Molecular Biology and Evolution*, *31*(6), 1402–1413. https://doi.org/10.1093/molbev/msu083

Chen, W.; Jia, Q.; Song, Y.; Fu, H.; Wei, G., & Ni, T. (2017). Alternative polyadenylation: Methods, findings, and impacts. *Genomics, Proteomics & Bioinformatics*, *15*(5), 287–300. https://doi.org/10.1016/j.gpb.2017.06.001

Chicco, D. (2017). Ten quick tips for machine learning in computational biology. *BioData Mining*, *10*, 35. https://doi.org/10.1186/s13040-017-0155-3

Cocquet, J.; Chong, A.; Zhang, G., & Veitia, R. A. (2006). Reverse transcriptase template switching and false alternative transcripts. *Genomics*, *88*(1), 127–131. https://doi.org/10.1016/j.ygeno.2005.12.013

Croucher, N. J.; Fookes, M. C.; Perkins, T. T.; Turner, D. J.; Marguerat, S. B.; Keane, T.; Quail, M. A.; He, M.; Assefa, S.; Bähler, J.; Kingsley, R. A.; Parkhill, J.; Bentley, S. D.; Dougan, G., & Thomson, N. R. (2009). A simple

method for directional transcriptome sequencing using illumina technology. *Nucleic Acids Research*, *37*(22), e148. https://doi.org/10.1093/nar/gkp811

de la Fuente, L.; Arzalluz-Luque, Á.; Tardáguila, M.; del Risco, C., Héctorand Martí; Tarazona, S.; Salguero, P.; Scott, R.; Lerma, A.; Alastrue-Agudo, A.; Bonilla, P.; Newman, J. R. B.; Kosugi, S.; McIntyre, L. M.; Moreno-Manzano, V., & Conesa, A. (2020). tappAS: A comprehensive computational framework for the analysis of the functional impact of differential splicing. *Genome Biology*, *21*(1), 119. https://doi.org/10.1186/s13059-020-02028-w

Dijk, E. L. v.; Jaszczyszyn, Y.; Naquin, D., & Thermes, C. (2018). The third revolution in sequencing technology [Publisher: Elsevier]. *Trends in Genetics*, *34*(9), 666–681. https://doi.org/10.1016/j.tig.2018.05.008

Eid, J.; Fehr, A.; Gray, J.; Luong, K.; Lyle, J.; Otto, G.; Peluso, P.; Rank, D.; Baybayan, P.; Bettman, B.; Bibillo, A.; Bjornson, K.; Chaudhuri, B.; Christians, F.; Cicero, R.; Clark, S.; Dalal, R.; deWinter, A.; Dixon, J., ... Turner, S. (2009). Real-time DNA sequencing from single polymerase molecules [Publisher: American Association for the Advancement of Science]. *Science*, *323*(5910), 133–138. https://doi.org/10.1126/science.1162986

ENCODE long read RNA-seq pipeline [original-date: 2019-02-04T17:21:17Z]. (2022, May 26). ENCODE DCC. Retrieved July 1, 2022, from https://github.com/ENCODE-DCC/long-read-rna-pipeline

Engström, P. G.; Steijger, T.; Sipos, B.; Grant, G. R.; Kahles, A.; Rätsch, G.; Goldman, N.; Hubbard, T. J.; Harrow, J.; Guigó, R., & Bertone, P. (2013). Systematic evaluation of spliced alignment programs for RNA-seq data. *Nature Methods*, *10*(12), 1185–1191. https://doi.org/10.1038/nmeth.2722

Feng, D., & Xie, J. (2013). Aberrant splicing in neurological diseases. *Wiley interdisciplinary reviews. RNA*, *4*(6), 631–649. https://doi.org/10.1002/wrna.1184

Forrest, A. R. R.; Kawaji, H.; Rehli, M.; Kenneth Baillie, J.; de Hoon, M. J. L.; Haberle, V.; Lassmann, T.; Kulakovskiy, I. V.; Lizio, M.; Itoh, M.; Andersson, R.; Mungall, C. J.; Meehan, T. F.; Schmeier, S.; Bertin, N.; Jørgensen, M.; Dimont, E.; Arner, E.; Schmidl, C., ... The FANTOM Consortium and the RIKEN PMI and CLST (DGT). (2014). A promoter-level mammalian expression atlas [Number: 7493 Publisher: Nature Publishing Group]. *Nature*, *507*(7493), 462–470. https://doi.org/10.1038/nature13182

Frankish, A.; Mudge, J. M.; Thomas, M., & Harrow, J. (2012). The importance of identifying alternative splicing in vertebrate genome annotation. *Database: The Journal of Biological Databases and Curation*, *2012*, bas014. https://doi.org/10.1093/database/bas014

Frankiw, L.; Baltimore, D., & Li, G. (2019). Alternative mRNA splicing in cancer immunotherapy. *Nature Reviews. Immunology*, *19*(11), 675–687. https://doi.org/10.1038/s41577-019-0195-7

Gallego Romero, I.; Pai, A. A.; Tung, J., & Gilad, Y. (2014). RNA-seq: Impact of RNA degradation on transcript quantification. *BMC Biology*, *12*(1), 42. https://doi.org/10.1186/1741-7007-12-42

Garneau, N. L.; Wilusz, J., & Wilusz, C. J. (2007). The highways and byways of mRNA decay. *Nature Reviews. Molecular Cell Biology*, *8*(2), 113–126. https://doi.org/10.1038/nrm2104

Goodwin, S.; McPherson, J. D., & McCombie, W. R. (2016). Coming of age: Ten years of next-generation sequencing technologies [Number: 6 Publisher: Nature Publishing Group]. *Nature Reviews Genetics*, *17*(6), 333–351. https://doi.org/10.1038/nrg.2016.49

jmestret. (2022, May 22). *Jmestret/SQANTISIM* [original-date: 2022-01-21T16:56:30Z]. Retrieved July 1, 2022, from https://github.com/jmestret/SQANTISIM

Kumar, A.; Clerici, M.; Muckenfuss, L. M.; Passmore, L. A., & Jinek, M. (2019). Mechanistic insights into mRNA 3-end processing. *Current Opinion in Structural Biology*, *59*, 143–150. https://doi.org/10.1016/j.sbi.2019.08.001

Leung, S. K.; Jeffries, A. R.; Castanho, I.; Jordan, B. T.; Moore, K.; Davies, J. P.; Dempster, E. L.; Bray, N. J.; O'Neill, P.; Tseng, E.; Ahmed, Z.; Collier, D. A.; Jeffery, E. D.; Prabhakar, S.; Schalkwyk, L.; Jops, C.; Gandal, M. J.; Sheynkman, G. M.; Hannon, E., & Mill, J. (2021). Full-length transcript sequencing of human and mouse cerebral cortex identifies widespread isoform diversity and alternative splicing. *Cell Reports*, *37*(7), 110022. https://doi.org/10.1016/j.celrep.2021.110022

Lexogen. (2017, May 30). *SIRV-sets | spike-in RNA variant control mixes | lexogen*. Retrieved July 1, 2022, from https://www.lexogen.com/sirvs/sirv-sets/

Minoche, A. E.; Dohm, J. C., & Himmelbauer, H. (2011). Evaluation of genomic high-throughput sequencing data generated on illumina HiSeq and genome analyzer systems. *Genome Biology*, *12*(11), R112. https://doi.org/10.1186/gb-2011-12-11-r112

Moore, J. E.; Purcaro, M. J.; Pratt, H. E.; Epstein, C. B.; Shoresh, N.; Adrian, J.; Kawli, T.; Davis, C. A.; Dobin, A.; Kaul, R.; Halow, J.; Van Nostrand, E. L.; Freese, P.; Gorkin, D. U.; Shen, Y.; He, Y.; Mackiewicz, M.; Pauli-Behn, F.; Williams, B. A., ... Weng, Z. (2020). Expanded encyclopaedias of DNA elements in the human and mouse genomes [Number: 7818 Publisher: Nature Publishing Group]. *Nature*, *583*(7818), 699–710. https://doi.org/10.1038/s41586-020-2493-4

Mudge, J. M.; Frankish, A.; Fernandez-Banet, J.; Alioto, T.; Derrien, T.; Howald, C.; Reymond, A.; Guigó, R.; Hubbard, T., & Harrow, J. (2011). The origins, evolution, and functional potential of alternative splicing in vertebrates. *Molecular Biology and Evolution*, *28*(10), 2949–2959. https://doi.org/10.1093/molbev/msr127

Nam, D. K.; Lee, S.; Zhou, G.; Cao, X.; Wang, C.; Clark, T.; Chen, J.; Rowley, J. D., & Wang, S. M. (2002). Oligo(dT) primer generates a high frequency of truncated cDNAs through internal poly(a) priming during reverse transcription. *Proceedings of the National Academy of Sciences of the United States of America*, *99*(9), 6152–6156. https://doi.org/10.1073/pnas.092140899

Newman, A. (1998). RNA splicing [Publisher: Elsevier]. *Current Biology*, *8*(25), R903–R905. https://doi.org/10.1016/S0960-9822(98)00005-0

Nilsen, T. W., & Graveley, B. R. (2010). Expansion of the eukaryotic proteome by alternative splicing. *Nature*, *463*(7280), 457–463. https://doi.org/10.1038/nature08909

Oikonomopoulos, S.; Wang, Y. C.; Djambazian, H.; Badescu, D., & Ragoussis, J. (2016). Benchmarking of the oxford nanopore MinION sequencing for quantitative and qualitative assessment of cDNA populations. *Scientific Reports*, *6*, 31602. https://doi.org/10.1038/srep31602

PacificBiosciences/IsoSeq [original-date: 2018-03-05T13:35:47Z]. (2022, May 12). PacBio. Retrieved May 24, 2022, from https://github.com/PacificBiosciences/IsoSeq

Park, J. W.; Fu, S.; Huang, B., & Xu, R.-H. (2020). Alternative splicing in mesenchymal stem cell differentiation. *Stem Cells (Dayton, Ohio)*, *38*(10), 1229–1240. https://doi.org/10.1002/stem.3248

R Core Team. (2020). *R: A language and environment for statistical computing*. R Foundation for Statistical Computing. Vienna, Austria. https://www.R-project.org/

Ramanathan, A.; Robb, G. B., & Chan, S.-H. (2016). mRNA capping: Biological functions and applications. *Nucleic Acids Research*, *44*(16), 7511–7526. https://doi.org/10.1093/nar/gkw551

Ren, F.; Zhang, N.; Zhang, L.; Miller, E., & Pu, J. J. (2020). Alternative polyadenylation: A new frontier in post transcriptional regulation. *Biomarker Research*, *8*(1), 67. https://doi.org/10.1186/s40364-020-00249-6

Ren, P.; Lu, L.; Cai, S.; Chen, J.; Lin, W., & Han, F. (2021). Alternative splicing: A new cause and potential therapeutic target in autoimmune disease. *Frontiers in Immunology*, *12*, 713540. https://doi.org/10.3389/fimmu.2021.713540

Rodríguez, S. A.; Grochová, D.; McKenna, T.; Borate, B.; Trivedi, N. S.; Erdos, M. R., & Eriksson, M. (2016). Global genome splicing analysis reveals an increased number of alternatively spliced genes with aging. *Aging Cell*, *15*(2), 267–278. https://doi.org/10.1111/acel.12433

Sharon, D.; Tilgner, H.; Grubert, F., & Snyder, M. (2013). A single-molecule long-read survey of the human transcriptome. *Nature biotechnology*, *31*(11), 1009–1014. https://doi.org/10.1038/nbt.2705

Shi, H.; Zhou, Y.; Jia, E.; Pan, M.; Bai, Y., & Ge, Q. (2021). Bias in RNA-seq library preparation: Current challenges and solutions. *BioMed Research International*, *2021*, 6647597. https://doi.org/10.1155/2021/6647597

Shkreta, L.; Bell, B.; Revil, T.; Venables, J. P.; Prinos, P.; Elela, S. A., & Chabot, B. (2013). Cancer-associated perturbations in alternative pre-messenger RNA splicing. *Cancer Treatment and Research*, *158*, 41–94. https://doi.org/10.1007/978-3-642-31659-3_3

Song, H.; Wang, L.; Chen, D., & Li, F. (2020). The function of pre-mRNA alternative splicing in mammal spermatogenesis. *International Journal of Biological Sciences*, *16*(1), 38–48. https://doi.org/10.7150/ijbs.34422

SQANTI3 [original-date: 2020-05-11T10:52:56Z]. (2022, May 8). ConesaLab - Genomics of gene expression. Retrieved May 26, 2022, from https://github.com/ConesaLab/SQANTI3

Steijger, T.; Abril, J. F.; Engström, P. G.; Kokocinski, F.; Hubbard, T. J.; Guigó, R.; Harrow, J., & Bertone, P. (2013). Assessment of transcript reconstruction methods for RNA-seq. *Nature Methods*, *10*(12), 1177–1184. https://doi.org/10.1038/nmeth.2714

TALON [original-date: 2018-04-03T20:07:02Z]. (2022, May 11). mortazavilab. Retrieved May 24, 2022, from https://github.com/mortazavilab/TALON

Tang, S.; Lomsadze, A., & Borodovsky, M. (2015). Identification of protein coding regions in RNA transcripts. *Nucleic Acids Research*, *43*(12), e78–e78. https://doi.org/10.1093/nar/gkv227

Tarca, A. L.; Carey, V. J.; Chen, X.-w.; Romero, R., & Drăghici, S. (2007). Machine learning and its applications to biology. *PLoS Computational Biology*, *3*(6), e116. https://doi.org/10.1371/journal.pcbi.0030116

Tardaguila, M.; de la Fuente, L.; Marti, C.; Pereira, C.; Pardo-Palacios, F. J.; del Risco, H.; Ferrell, M.; Mellado, M.; Macchietto, M.; Verheggen, K.; Edelmann, M.; Ezkurdia, I.; Vazquez, J.; Tress, M.; Mortazavi, A.; Martens, L.; Rodriguez-Navarro, S.; Moreno-Manzano, V., & Conesa, A. (2018). SQANTI: Extensive characterization of long-read transcript sequences for quality control in full-length transcriptome identification and quantification. *Genome Research*, *28*(3), 396–411. https://doi.org/10.1101/gr.222976.117

Teichroeb, J. H.; Kim, J., & Betts, D. H. (2016). The role of telomeres and telomerase reverse transcriptase isoforms in pluripotency induction and maintenance. *RNA Biology*, *13*(8), 707–719. https://doi.org/10.1080/15476286.2015.1134413

Tian, B.; Hu, J.; Zhang, H., & Lutz, C. S. (2005). A large-scale analysis of mRNA polyadenylation of human and mouse genes. *Nucleic Acids Research*, *33*(1), 201–212. https://doi.org/10.1093/nar/gki158

Travers, K. J.; Chin, C.-S.; Rank, D. R.; Eid, J. S., & Turner, S. W. (2010). A flexible and efficient template format for circular consensus sequencing and SNP detection. *Nucleic Acids Research*, *38*(15), e159. https://doi.org/10.1093/nar/gkq543

Tseng, E. (2022, May 27). *cDNA_cupcake* [original-date: 2016-05-09T20:08:59Z]. Retrieved May 31, 2022, from https://github.com/Magdoll/cDNA_Cupcake

Wang, R.; Zheng, D.; Yehia, G., & Tian, B. (2018). A compendium of conserved cleavage and polyadenylation events in mammalian genes. *Genome Research*, *28*(10), 1427–1441. https://doi.org/10.1101/gr.237826.118

Wang, Y.; Xu, Y., & Zhang, Y. (2022). A new signature based on alternative polyadenylation for prognostic prediction and therapeutic responses in low-grade glioma. *Aging (Albany NY)*, *14*(2), 826–844. https://doi.org/10.18632/aging.203844

Wang, Z.; Gerstein, M., & Snyder, M. (2009). RNA-seq: A revolutionary tool for transcriptomics. *Nature reviews. Genetics*, *10*(1), 57–63. https://doi.org/10.1038/nrg2484

Wickham, H.; Averick, M.; Bryan, J.; Chang, W.; McGowan, L. D.; François, R.; Grolemund, G.; Hayes, A.; Henry, L.; Hester, J.; Kuhn, M.; Pedersen, T. L.; Miller, E.; Bache, S. M.; Müller, K.; Ooms, J.; Robinson, D.; Seidel, D. P.; Spinu, V., ... Yutani, H. (2019). Welcome to the tidyverse. *Journal of Open Source Software*, *4*(43), 1686. https://doi.org/10.21105/joss.01686

Wyman, D.; Balderrama-Gutierrez, G.; Reese, F.; Jiang, S.; Rahmanian, S.; Forner, S.; Matheos, D.; Zeng, W.; Williams, B.; Trout, D.; England, W.; Chu, S.-H.; Spitale, R. C.; Tenner, A. J.; Wold, B. J., & Mortazavi, A. (2020, March 24). A technology-agnostic long-read analysis pipeline for transcriptome

discovery and quantification [Pages: 672931 Section: New Results]. https://doi.org/10.1101/672931

Yeh, H.-S., & Yong, J. (2016). Alternative polyadenylation of mRNAs: 3-untranslated region matters in gene expression. *Molecules and Cells*, *39*(4), 281–285. https://doi.org/10.14348/molcells.2016.0035

Zhang, Y.; Liu, L.; Qiu, Q.; Zhou, Q.; Ding, J.; Lu, Y., & Liu, P. (2021). Alternative polyadenylation: Methods, mechanism, function, and role in cancer. *Journal of experimental & clinical cancer research: CR*, *40*(1), 51. https://doi.org/10.1186/s13046-021-01852-7

Zhang, Y.; Xu, Y., & Wang, Y. (2022). Alternative polyadenylation associated with prognosis and therapy in colorectal cancer. *Scientific Reports*, *12*(1), 7036. https://doi.org/10.1038/s41598-022-11089-9

Appendix A

R script for analysing MLfilter output

SAMPLE R SCRIPT

Generation diagnostic plots for IsoSeq3 processed data

Eva Estevan Morió

Contents

First steps

Loading necessary packages

```
library(tidyverse)
library(cowplot)
library(scales)
library(RColorConesa)
library(UpSetR)
library(optparse)
library(funprog)
library(ggplotify)
library(ggrepel)
```

Setting theme parameters

```
sq_theme <- theme_classic(base_family = "Helvetica") +
  theme(plot.title = element_text(lineheight=.4, size=15, hjust = 0.5),
        plot.subtitle = element_text(hjust = 0.5)) +
  theme(plot.margin = unit(c(2.5,1,1,1), "cm")) +
  theme(axis.line.x = element_line(color="black", size = 0.4),
        axis.line.y = element_line(color="black", size = 0.4)) +
  theme(axis.title.x = element_text(size=14),
        axis.text.x  = element_text(size=14),
        axis.title.y = element_text(size=14),
        axis.text.y  = element_text(vjust=0.5, size=13) ) +
  theme(legend.text = element_text(size = 12),
        legend.title = element_text(size=14),
        legend.key.size = unit(0.5, "cm"))


theme_set(sq_theme)
```

Setting colors for each filtering combination: FSM FSM_dist RM RM_dist

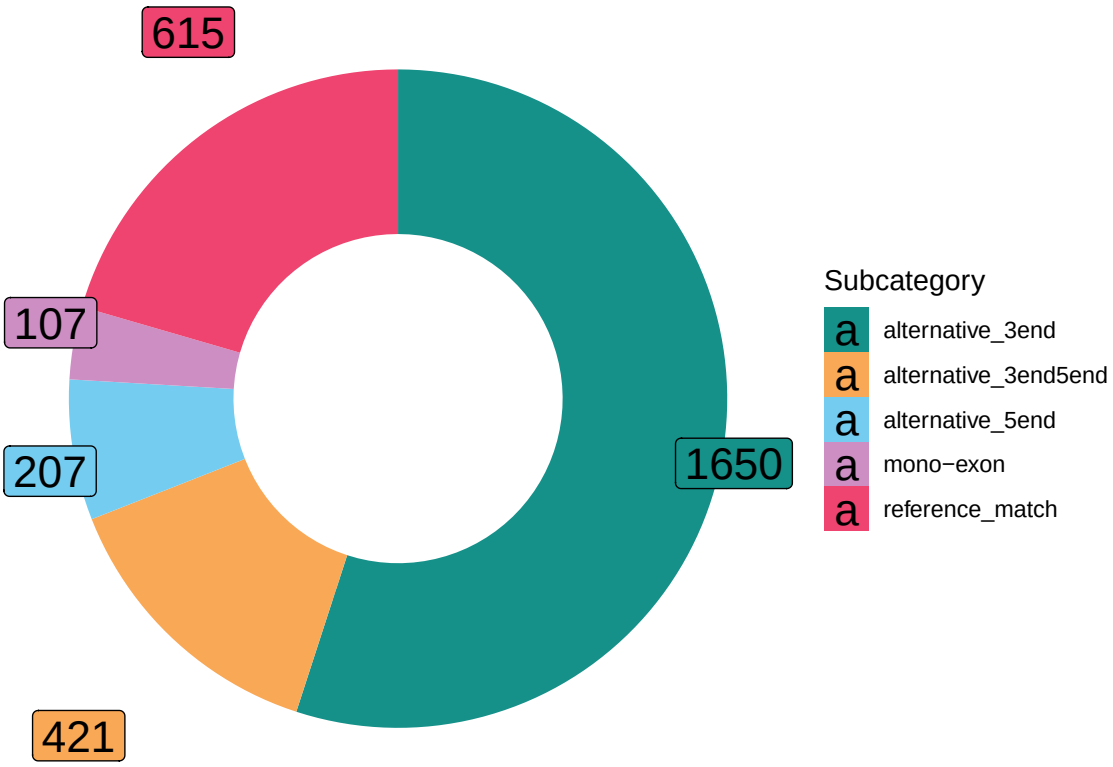
```
filt_palette=c(IsoSeq_RM ="lightslateblue", IsoSeq_FSM="darkorange",IsoSeq_RM_dist="skyblue",
               IsoSeq_FSM_dist="gold",  Common="darkcyan")


filt_palette_un=c(Unique_IsoSeq_RM ="lightslateblue", Unique_IsoSeq_FSM="darkorange"
               ,Unique_IsoSeq_RM_dist="skyblue",
               Unique_IsoSeq_FSM_dist="gold",  Common="darkcyan")
```

Isoseq3 data analysis

TP FSM subcategories

First, the TP FSM set file is loaded to generate a pie plot with the percentages of the subcategories.

```
#load TP FSM file
TP_GM<-read.csv('SQ3_MLresults/GM12878/GM12878_input/TP_FSM_GM12878.txt', header=FALSE)

#select de isoform
colnames(TP_GM)<-'isoform'

#load the input classification file
classif_GM12878<-read_tsv('SQ3_MLresults/GM12878/GM12878_input/GM12878_classification.txt')

#select the columns isoform and subcategory
classif_GM12878<-classif_GM12878 %>% select(isoform,subcategory)

#intersect both tables
TP_GM<-left_join(TP_GM,classif_GM12878)

#calculate percentages of each subcategory
cat<-TP_GM %>% group_by(subcategory) %>% summarize(category_count = n()) %>%
  mutate(percentin=category_count/sum(category_count)) %>%
  mutate(suma=cumsum(category_count))

# Compute percentages
cat$fraction <- cat$category_count / sum(cat$category_count)

# Compute the cumulative percentages (top of each rectangle)
cat$ymax <- cumsum(cat$fraction)

# Compute the bottom of each rectangle
cat$ymin <- c(0, head(cat$ymax, n=-1))

#label positions
cat$labelPosition <- (cat$ymax + cat$ymin) / 2

# Compute a good label
cat$label <- paste0(cat$category_count)

#PIE PLOT
tp_pie<- ggplot(cat, aes(ymax=ymax, ymin=ymin, xmax=4, xmin=3, fill=subcategory)) +
  geom_rect()+
  RColorConesa::scale_fill_conesa(palette = "complete")+
  coord_polar(theta="y") +
  xlim(c(2, 4)) +
  geom_label_repel(data = cat,x=4.5, aes(y=labelPosition, label=label), size=6) +
  guides(fill = guide_legend(title = "Subcategory")) +
  theme_void()
```


615
107
207
421
1650
Subcategory
alternative_3end
alternative_3end5end
alternative_5end
mono−exon
reference_match

Loading output classification files

Then, the four classification files generated by the Machine Learning filter are loaded. For this purpose, each one is given a label.

```
# first output to compare
path1 <- "SQ3_MLresults/GM12878/GM12878_output_withcol"
label1 <- "IsoSeq_RM_dist"
files1 <- dir(path1)

# second output to compare
path2 <- "SQ3_MLresults/GM12878/GM12878_output_FSM_withcol"
label2 <- "IsoSeq_FSM_dist"
files2 <- dir(path2)

# third output to compare
path3 <- "SQ3_MLresults/GM12878/GM12878_output_FSM"
label3 <- "IsoSeq_FSM"
files3 <- dir(path3)

# fourth output to compare
path4 <- "SQ3_MLresults/GM12878/GM12878_output"
label4 <- "IsoSeq_RM"
files4 <- dir(path4)


#Load first classification files
classif1_file <- files1[str_detect(files1, "MLresult_classification")]
classif_1 <- read_tsv(paste0(path1, "/", classif1_file))

classif_1 <- classif_1 %>%
  mutate(structural_category = factor(structural_category) %>%
           fct_infreq() %>%
           fct_recode(ISM = "incomplete-splice_match",
                      FSM = "full-splice_match",
                      NNC = "novel_not_in_catalog",
                      NIC = "novel_in_catalog",
                      Intergenic = "intergenic",
                      Antisense = "antisense",
                      Genic = "genic",
                      Fusion = "fusion",
                      Genic_intron = "genic_intron") %>%
           fct_relevel(c("FSM", "ISM", "NIC", "NNC",
                         "Genic", "Antisense", "Fusion",
                         "Intergenic", "Genic_intron")))


# Load second classification file
classif2_file <- files2[str_detect(files2, "MLresult_classification")]
classif_2 <- read_tsv(paste0( path2, "/", classif2_file))

classif_2 <- classif_2 %>%
  mutate(structural_category = factor(structural_category) %>%
           fct_infreq() %>%
```

```
          fct_recode(ISM = "incomplete-splice_match",
                     FSM = "full-splice_match",
                     NNC = "novel_not_in_catalog",
                     NIC = "novel_in_catalog",
                     Intergenic = "intergenic",
                     Antisense = "antisense",
                     Genic = "genic",
                     Fusion = "fusion",
                     Genic_intron = "genic_intron") %>%
          fct_relevel(c("FSM", "ISM", "NIC", "NNC",
                        "Genic", "Antisense", "Fusion",
                        "Intergenic", "Genic_intron")))


# Load third classification file
classif3_file <- files3[str_detect(files3, "MLresult_classification")]
classif_3 <- read_tsv(paste0(path3, "/", classif3_file))

classif_3 <- classif_3 %>%
  mutate(structural_category = factor(structural_category) %>%
          fct_infreq() %>%
          fct_recode(ISM = "incomplete-splice_match",
                     FSM = "full-splice_match",
                     NNC = "novel_not_in_catalog",
                     NIC = "novel_in_catalog",
                     Intergenic = "intergenic",
                     Antisense = "antisense",
                     Genic = "genic",
                     Fusion = "fusion",
                     Genic_intron = "genic_intron") %>%
          fct_relevel(c("FSM", "ISM", "NIC", "NNC",
                        "Genic", "Antisense", "Fusion",
                        "Intergenic", "Genic_intron")))

# Load fourth classification file
classif4_file <- files4[str_detect(files4, "MLresult_classification")]
classif_4 <- read_tsv(paste0(path4, "/", classif4_file))

classif_4 <- classif_4 %>%
  mutate(structural_category = factor(structural_category) %>%
          fct_infreq() %>%
          fct_recode(ISM = "incomplete-splice_match",
                     FSM = "full-splice_match",
                     NNC = "novel_not_in_catalog",
                     NIC = "novel_in_catalog",
                     Intergenic = "intergenic",
                     Antisense = "antisense",
                     Genic = "genic",
                     Fusion = "fusion",
                     Genic_intron = "genic_intron") %>%
          fct_relevel(c("FSM", "ISM", "NIC", "NNC",
                        "Genic", "Antisense", "Fusion",
                        "Intergenic", "Genic_intron")))
```

Variable importance plot

Importance files (in which the values of importance of each variable are stored), are loaded.

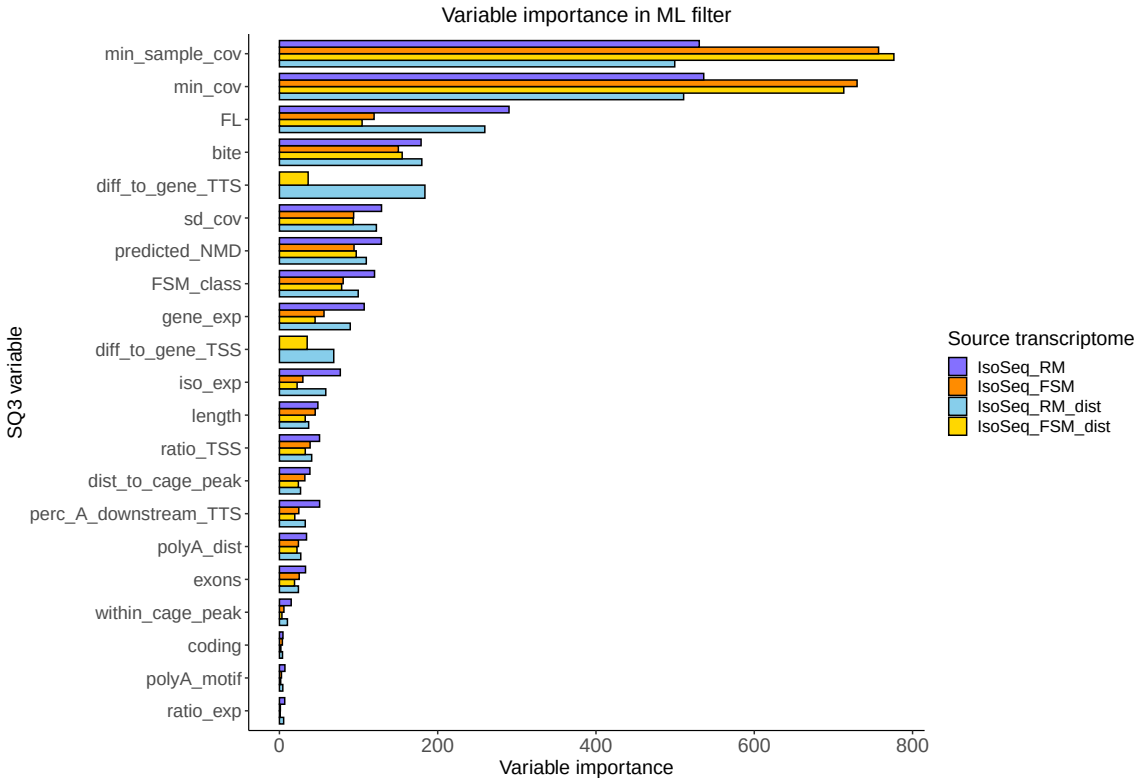
```
imp1_file <- files1[str_detect(files1, "variable-importance")]
imp_1 <- read_tsv(paste0( path1, "/", imp1_file),
                  col_names = c("variable", "importance"))


imp2_file <- files2[str_detect(files2, "variable-importance")]
imp_2 <- read_tsv(paste0( path2, "/", imp2_file),
                  col_names = c("variable", "importance"))

imp3_file <- files3[str_detect(files3, "variable-importance")]
imp_3 <- read_tsv(paste0( path3, "/", imp3_file),
                  col_names = c("variable", "importance"))

imp4_file <- files4[str_detect(files4, "variable-importance")]
imp_4 <- read_tsv(paste0( path4, "/", imp4_file),
                  col_names = c("variable", "importance"))
```

The four importance tables are now combined.

```
imp_combined <- bind_rows(list(imp_1 = imp_1, imp_2 = imp_2,
                               imp_3 = imp_3, imp_4 = imp_4),
                          .id = "source_transcriptome")

imp_combined <- imp_combined %>%
  mutate(variable = fct_reorder(variable, importance),
         source_transcriptome = factor(source_transcriptome,
                                       levels = c("imp_1", "imp_2",
                                                  "imp_3", "imp_4"),
                                       labels = c(label1, label2,
                                                  label3, label4)))
```

And the plot is generated.

```
imp_plot <- ggplot(imp_combined,
                   aes(x = variable, y = importance, fill=source_transcriptome)) +
  ggtitle("Variable importance in ML filter") +
  geom_bar(width = 0.8, color = "black",
           stat = "identity", position = "dodge") +
  labs(x = "SQ3 variable", y = "Variable importance") +
  scale_fill_manual(values = filt_palette[1:4], name="Source transcriptome") +
  coord_flip()
```


Variable importance in ML filter
min_sample_cov
min_cov
FL
bite
diff_to_gene_TTS
sd_cov
predicted_NMD
FSM_class
gene_exp
diff_to_gene_TSS
iso_exp
length
ratio_TSS
dist_to_cage_peak
perc_A_downstream_TTS
polyA_dist
exons
within_cage_peak
coding
polyA_motif
ratio_exp
SQ3 variable
0
200
400
600
800
Variable importance
Source transcriptome
IsoSeq_RM
IsoSeq_FSM
IsoSeq_RM_dist
IsoSeq_FSM_dist

Number of artifacts

First, we filter the transcripts considered artifacts in each classification file and save them in a new table.

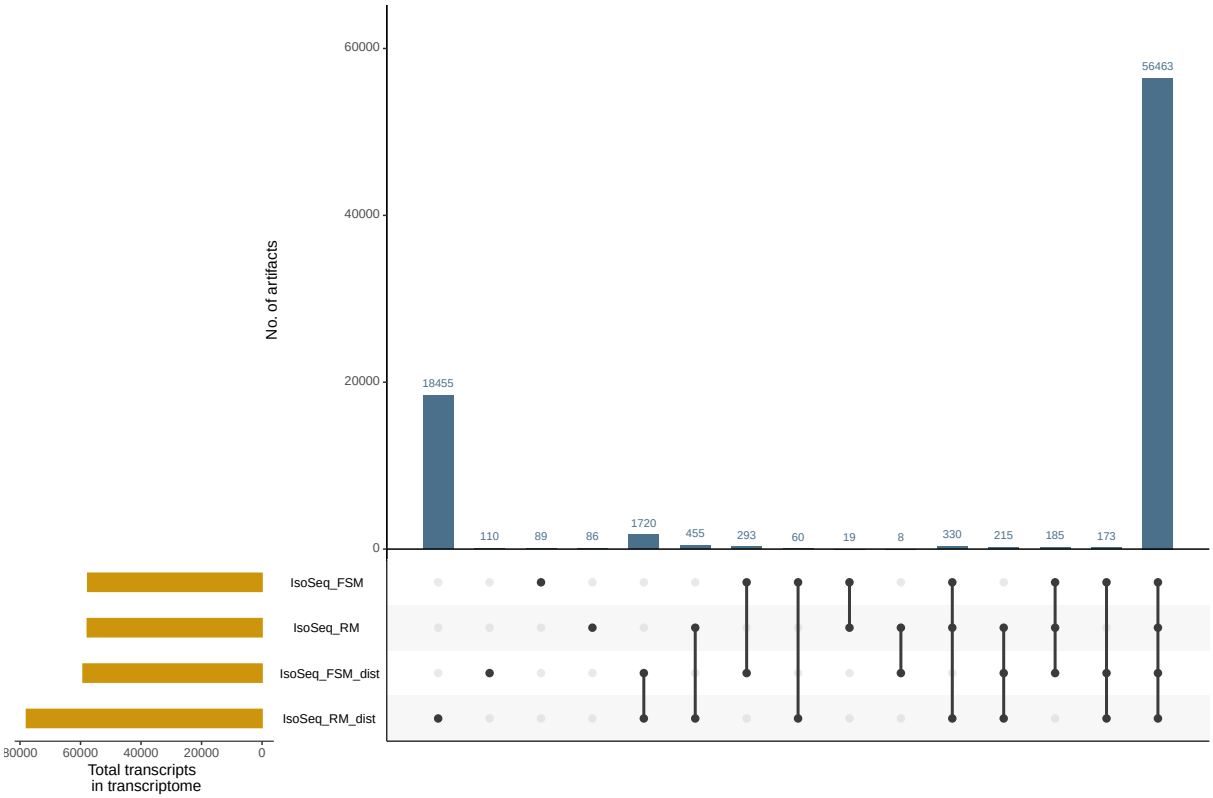
```
artifacts_1 <- filter(classif_1, filter_result == "Artifact")
artifacts_2 <- filter(classif_2, filter_result == "Artifact")
artifacts_3 <- filter(classif_3, filter_result == "Artifact")
artifacts_4 <- filter(classif_4, filter_result == "Artifact")
```

Then, we create a list of these artifacts (iso_list) and add the names of the filtering combination they come from.

```
iso_list <- list(classif_1 = artifacts_1 %>% select(isoform) %>% deframe,
                 classif_2 = artifacts_2 %>% select(isoform) %>% deframe,
                 classif_3 = artifacts_3 %>% select(isoform) %>% deframe,
                 classif_4 = artifacts_4 %>% select(isoform) %>% deframe)

names(iso_list) <- c(label1, label2, label3, label4)
```

Finally, we generate a plot to represent the intersections of the four filtering combination compared.

```
intersections <- fromList(iso_list)

upset <- upset(intersections, main.bar.color = "skyblue4",
      mainbar.y.label = "No. of artifacts",
      sets.bar.color = "darkgoldenrod3",
      sets.x.label = "Total transcripts \n in transcriptome",
      point.size = 2.5, line.size = 1, text.scale = c(1.3, 1.3, 1.3, 1.3, 1.3, 1.3))
```


No. of artifacts
60000
40000
20000
0
18455
110
89
86
1720
455
293
60
19
8
330
215
185
173
56463
IsoSeq_FSM
IsoSeq_RM
IsoSeq_FSM_dist
IsoSeq_RM_dist
80000
60000
40000
20000
0
Total transcripts
in transcriptome

IsoSeq3 first comparison: FSM_dist vs RM_dist

Categories plots

First, we create a list of artifacts of both filtering combinations (artifact_list) and add the name of the source transcriptome.

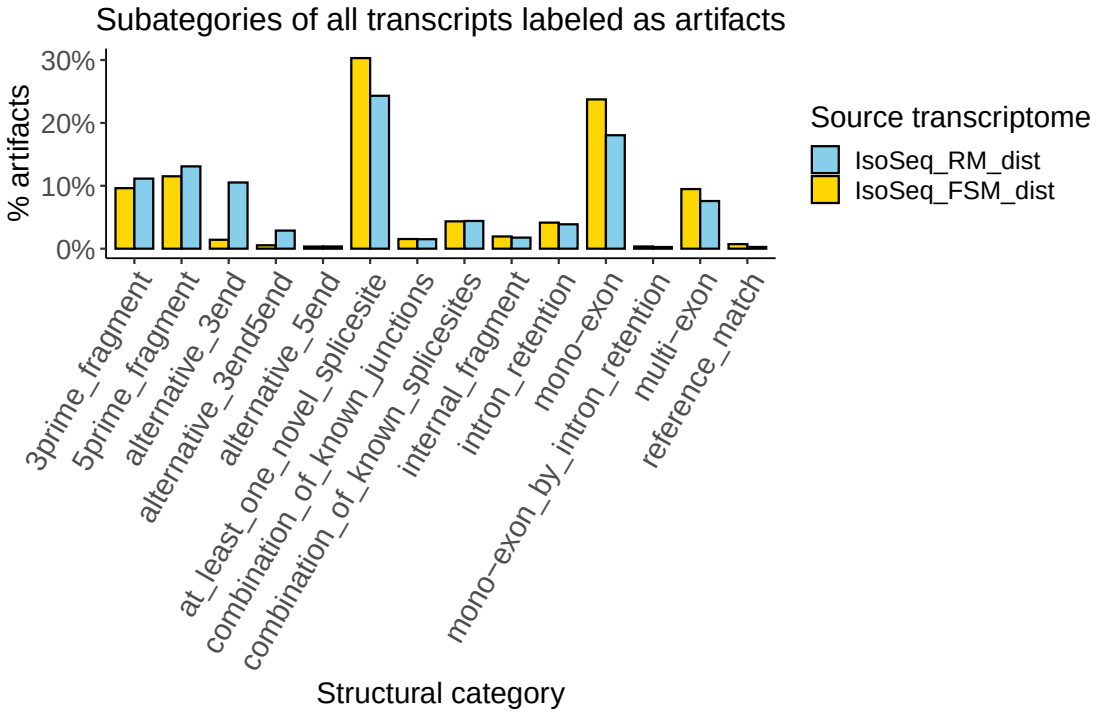
```
artifact_list <- list(artifacts_1, artifacts_2)
names(artifact_list) <- c(label1, label2)

artifacts_all <- bind_rows(artifact_list,
                           .id = "source_transcriptome")
```

Then we calculate the percentage of artifacts filtered by each combination for each subcategory (artifact_summary).

```
artifact_summary <- artifacts_all %>%
  group_by(source_transcriptome, subcategory) %>%
  summarize(category_count = n()) %>%
  mutate(percent = category_count/sum(category_count))
```

And plot it.

```
cat_all <- ggplot(artifact_summary,
                  aes(x = subcategory, y = percent)) +
  ggtitle("Subategories of all transcripts labeled as artifacts") +
  geom_bar(aes(fill = source_transcriptome), stat = "identity", position = "dodge",
           width = 0.8, color = "black") +
  labs(x = "Structural category", y = "% artifacts") +
  scale_y_continuous(labels = scales::percent_format()) +
  scale_fill_manual(values=filt_palette[3:4], name="Source transcriptome")+
  theme(axis.text.x = element_text(angle = 60, hjust = 1))
```



We extract those unique artifacts from each classification file and we make a list with all of them.

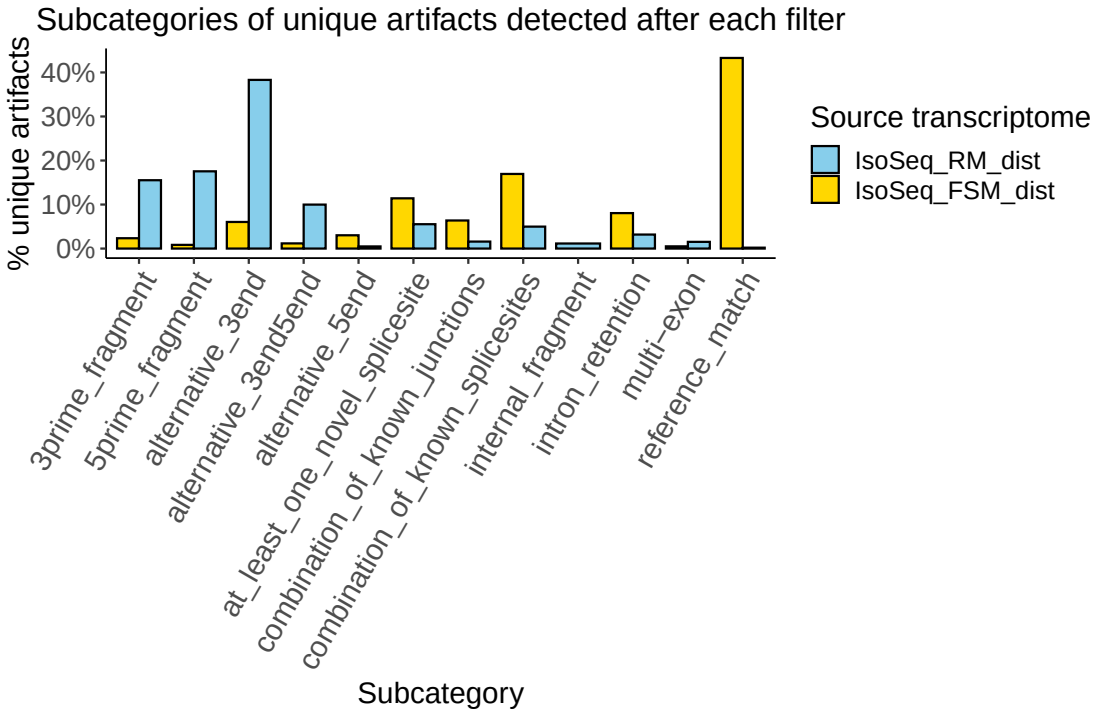
```
unique_1 <- filter(classif_1, filter_result == "Artifact" &
                     !(isoform %in% intersect(iso_list[[1]], iso_list[[2]])))

unique_2 <- filter(classif_2, filter_result == "Artifact" &
                     !(isoform %in% intersect(iso_list[[1]], iso_list[[2]])))

unique_list <- list(unique_1, unique_2)
names(unique_list) <- c(label1, label2)

artifacts_unique <- bind_rows(unique_list,
                              .id = "source_transcriptome")
```

And calculate the percentage of artifacts filtered by each combination (unique_summary).

```
unique_summary <- artifacts_unique %>%
  group_by(source_transcriptome, subcategory) %>%
  summarize(category_count = n()) %>%
  mutate(percent = category_count/sum(category_count))
```

To plot it.

```
cat_un <- ggplot(unique_summary,
                 aes(x = subcategory, y = percent,
                     fill = source_transcriptome)) +
  ggtitle("Subcategories of unique artifacts detected after each filter") +
  geom_bar(stat = "identity", position = "dodge",
           width = 0.8, color = "black") +
  labs(x = "Subcategory", y = "% unique artifacts") +
  scale_y_continuous(labels = percent_format()) +
  scale_fill_manual(values = filt_palette[3:4], name="Source transcriptome")+
    theme(axis.text.x = element_text(angle = 60, hjust = 1))
```



In order to see this information all together, with common and unique artifacts in one plot, we add a column to classification to artifacts_all list including whether the artifact is unique or common.

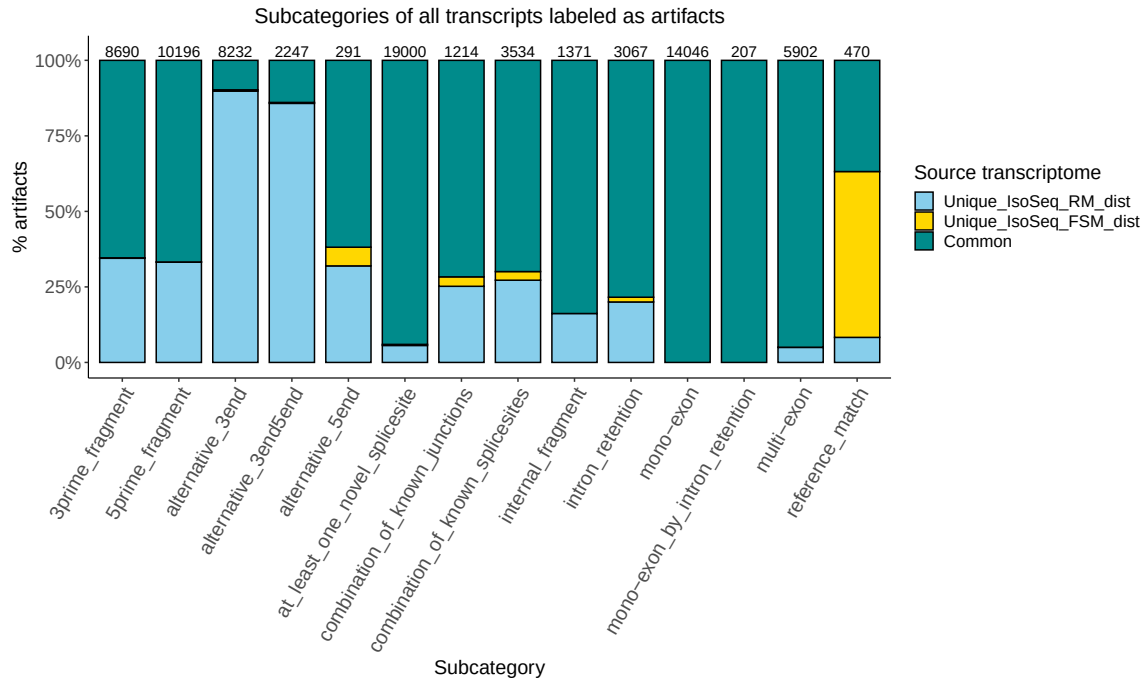
```
artifacts_all <- artifacts_all %>%
  mutate(artifact_type = case_when(isoform %in% artifacts_unique$isoform == TRUE ~ "Unique",
                                   isoform %in% artifacts_unique$isoform == FALSE ~ "Common"),
         artifact_lab = if_else(artifact_type == "Unique",
                                true = paste0(artifact_type, "_", source_transcriptome),
                                false = "Common"))
```

Then, we calculate again the percentages.

```
artifact_sum <- artifacts_all %>%
  dplyr::filter(!(source_transcriptome == label2 &
                    artifact_lab == 'Common')) %>%
  group_by(source_transcriptome,  subcategory, artifact_lab) %>%
  summarize(category_count = n()) %>%
  group_by(subcategory) %>%
  mutate(percent = category_count/sum(category_count),
         suma=cumsum(category_count)) %>%
  arrange(desc(artifact_lab))
```

To finally plot the percentages of categories of all transcripts labeled as artifacts.

```
cat_stack <- ggplot(artifact_sum,
                    aes(x =  subcategory, y = percent)) +
  ggtitle("Subcategories of all transcripts labeled as artifacts") +
  geom_bar(aes( fill =artifact_lab), stat = "identity", position = "stack",
           width = 0.8, color = "black") +
  labs(x = "Subcategory", y = "% artifacts") +
  scale_y_continuous(labels = scales::percent_format()) +
  scale_fill_manual(values = filt_palette_un[3:5], name="Source transcriptome")+
  geom_text(aes(x=subcategory,y=1.03, label=suma),size=4,
            check_overlap=TRUE)+
  theme(axis.text.x = element_text(angle = 60, hjust = 1))
```



Variables comparison plots

For comparing the values of each variable, we make a function (compare_artifacts), that takes the artifacts_all table, each variable and the imp_combined table to return a set of plots.

```
compare_artifacts <- function(classification,
                              var, importance){

  require(ggplot2)
  require(magrittr)

  # Select variable for evaluation plot
  var_df <- classification %>%
    dplyr::select(structural_category,
                  source_transcriptome,
                  artifact_type, artifact_lab,
                  dplyr::all_of(var))

  # Explicitly remove NAs
  var_df <- var_df %>%
    dplyr::filter(!(is.na(var)))

  # Get variable column info (class, name)
  var_type <- purrr::map_chr(var_df, class)
  var_name <- names(var_type[5])

  # Rename variable column to handle during plotting
  var_df <- var_df %>% dplyr::rename(variable = var)

  # Obtain labels
  labels <- importance %>%
    dplyr::filter(variable == var) %>%
    select(source_transcriptome) %>% deframe %>% levels()

  # Obtain and round importance
  imp <- importance %>%
    dplyr::filter(variable == var) %>%
    select(importance) %>% deframe
  imp <- round(imp, 2)

  # Generate plot by type
  if(var_type[5] == "numeric"){

    if(var_name=='min_sample_cov'){

      var_df <- var_df %>%
        dplyr::filter(!is.na(variable),
                      !(source_transcriptome == labels[2] &
                          artifact_lab == 'Common'))%>%
        dplyr::mutate(variable = factor(variable)) %>%
        group_by(source_transcriptome,
                 structural_category, artifact_lab, variable) %>%
        summarize(category_count = n()) %>%
        group_by(source_transcriptome, structural_category, artifact_lab) %>%
```

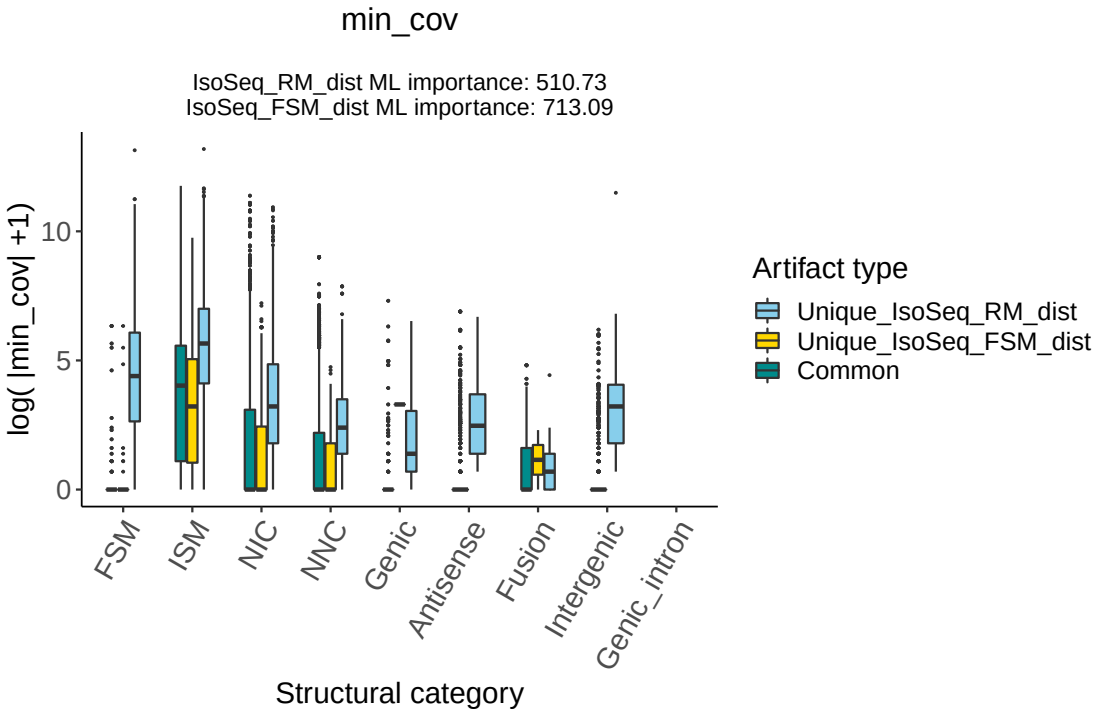
```
      mutate(percentin=category_count/sum(category_count)) %>%
      mutate(suma=cumsum(category_count)) %>%
      arrange(desc(variable))

    p<- ggplot(var_df) +
      ggtitle(paste(var, "\n"),
              subtitle = paste0("\n", labels[1], " ML importance: ", imp[1],
                                "\n", labels[2], " ML importance: ", imp[2])) +
      geom_bar(aes(y=percentin,x = artifact_lab, fill = variable ),
               stat = "identity", width = 0.8, color = "black",
               position = "stack") +
      labs(x = "Artifact type", y = "Percentage") +
      theme(axis.text.x = element_text(angle = 60, hjust = 1)) +
      geom_text(aes(x=artifact_lab,y=1.03, label=suma),size=2,
                check_overlap=TRUE)+
      scale_y_continuous(labels=scales::label_percent())+
              RColorConesa::scale_fill_conesa(paste0(var), palette = "complete",
                                      continuous = FALSE, reverse = FALSE) +
      facet_grid(~structural_category, scales = "free")


    return(p)

  }

  else{
    p <- ggplot(var_df) +
      ggtitle(paste(var),
              subtitle = paste0("\n", labels[1], " ML importance: ", imp[1],
                                "\n", labels[2], " ML importance: ", imp[2])) +
      geom_boxplot(aes(x = structural_category, y = log(abs(variable)+1),
                       fill = artifact_lab),
                   outlier.size = 0.2, width = 0.5) +
      labs(x = "Structural category", y = paste0("log( |", var_name, "| +1)")) +
      scale_fill_manual(values=filt_palette_un[3:5],name="Artifact type")+
      theme(axis.text.x = element_text(angle = 60, hjust = 1))


    return(p)

  }

} else if(var_type[5] == "integer"){

  # Specific plot for exon-related columns (integer variables divided into intervals)
  var_fct <- var_df %>%
    dplyr::filter(artifact_type == "Unique") %>%
    dplyr::mutate(variable = cut(variable, breaks = c(0, 1, 3, 5, 10, max(.$variable)),
                                 labels = c("1", "2-3", "4-5", "6-10", ">10")))

  p <- ggplot(var_fct) +
    ggtitle(paste(var, "\n"),
            subtitle = paste0("\n", labels[1], " ML importance: ", imp[1],
```

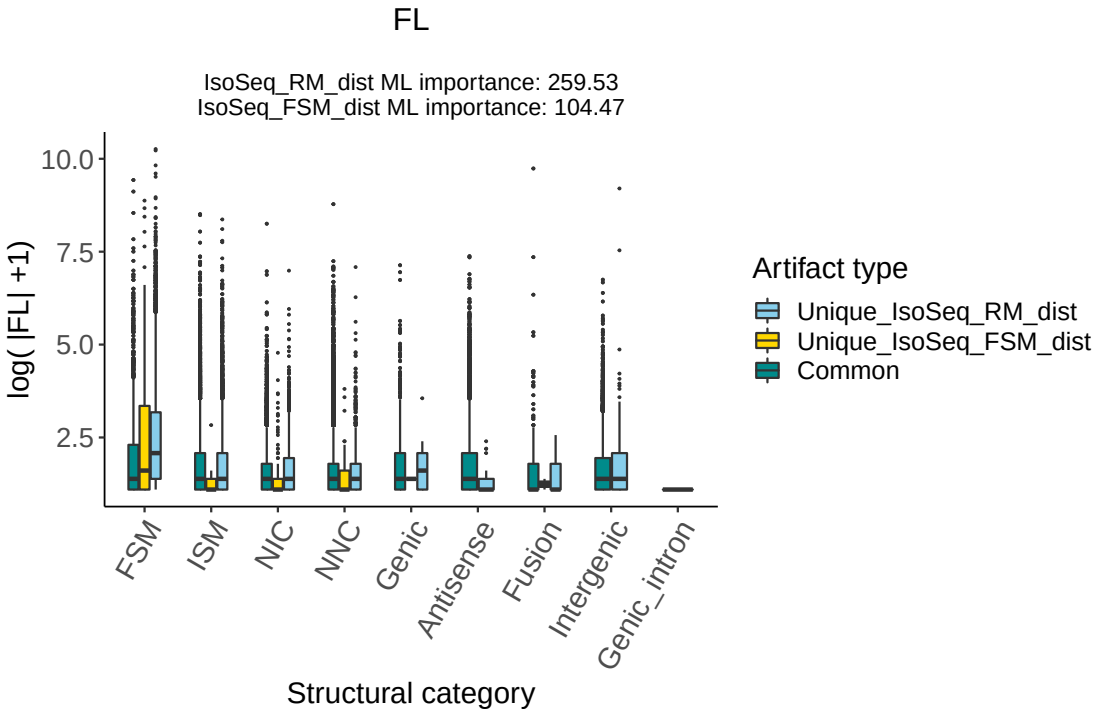
```
                                "\n", labels[2], " ML importance: ", imp[2], "\n")) +
      geom_bar(aes(x = artifact_lab, fill = variable), stat = "count",
               width = 0.8, color = "black", position = "dodge") +
      labs(x = "Artifact type \n(common artifacts not displayed)",
           y = "Transcript no.") +
      theme(axis.text.x = element_text(angle = 60, hjust = 1)) +
      scale_fill_manual(values=filt_palette_un[3:5],name=paste0(var))+
      facet_grid(~structural_category, scales = "free")

    return(p)

  } else{
    var_df <- var_df %>%
      dplyr::filter(!is.na(variable),
                    !(source_transcriptome == labels[2] &
                        artifact_lab == 'Common'))%>%
      dplyr::mutate(variable = factor(variable)) %>%
      group_by(source_transcriptome,
               structural_category, artifact_lab, variable) %>%
      summarize(category_count = n()) %>%
      group_by(source_transcriptome, structural_category, artifact_lab) %>%
      mutate(percentin=category_count/sum(category_count)) %>%
      mutate(suma=cumsum(category_count)) %>%
      arrange(desc(variable))

    p<- ggplot(var_df) +
      ggtitle(paste(var, "\n"),
              subtitle = paste0("\n", labels[1], " ML importance: ", imp[1],
                                "\n", labels[2], " ML importance: ", imp[2])) +
      geom_bar(aes(y=percentin,x = artifact_lab, fill = variable ),
               stat = "identity", width = 0.8, color = "black",
               position = "stack") +
      labs(x = "Artifact type", y = "Percentage") +
      theme(axis.text.x = element_text(angle = 60, hjust = 1)) +
      geom_text(aes(x=artifact_lab,y=1.03, label=suma),size=2,
                check_overlap=TRUE)+
      scale_y_continuous(labels=scales::label_percent())+
      scale_fill_manual(values=filt_palette_un[3:5],name=paste0(var))+
      facet_grid(~structural_category, scales = "free")


    return(p)

  }
}
```

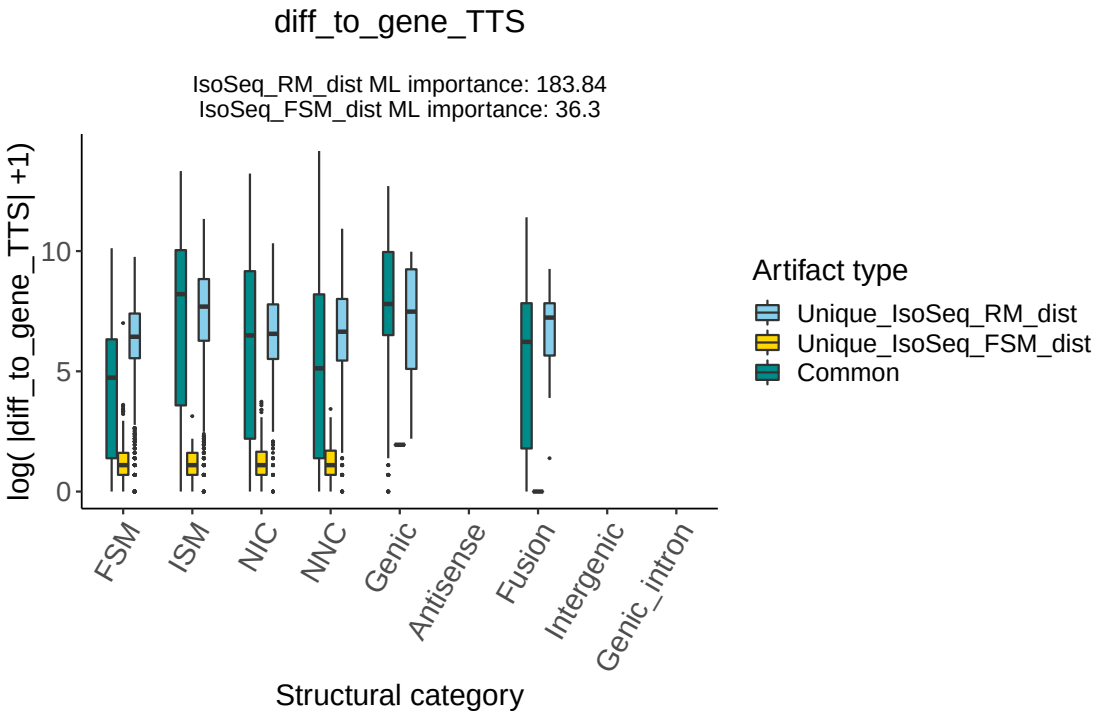
So that, the next code will generate the plots comparing the values of all the variables:

```
art_compare <- purrr::map(imp_combined$variable %>% unique,
                          ~compare_artifacts(artifacts_all, ., imp_combined))
```

As an example, it is shown some of the plots generated:


min_cov
IsoSeq_RM_dist ML importance: 510.73
IsoSeq_FSM_dist ML importance: 713.09
log(|min_cov| +1)
10
5
0
FSM
ISM
NIC
NNC
Genic
Antisense
Fusion
Intergenic
Genic_intron
Structural category
Artifact type
Unique_IsoSeq_RM_dist
Unique_IsoSeq_FSM_dist
Common


FL
IsoSeq_RM_dist ML importance: 259.53
IsoSeq_FSM_dist ML importance: 104.47
log(|FL| +1)
10.0
7.5
5.0
2.5
FSM
ISM
NIC
NNC
Genic
Antisense
Fusion
Intergenic
Genic_intron
Structural category
Artifact type
Unique_IsoSeq_RM_dist
Unique_IsoSeq_FSM_dist
Common


diff_to_gene_TTS
IsoSeq_RM_dist ML importance: 183.84
IsoSeq_FSM_dist ML importance: 36.3
log(|diff_to_gene_TTS| +1)
10
5
0
FSM
ISM
NIC
NNC
Genic
Antisense
Fusion
Intergenic
Genic_intron
Structural category
Artifact type
Unique_IsoSeq_RM_dist
Unique_IsoSeq_FSM_dist
Common

Bidimensional plots (for comparing each variable vs distance variables)

First of all, we store the names of the distance variables columns in distcol.

```
distcol<-colnames(artifacts_all)[str_detect(colnames(artifacts_all), "diff")]
```

And we also store the names of the structural categories in scat.

```
scat<- select(artifacts_all,structural_category) %>% filter(!duplicated(structural_category))
scat<-as.character(scat$structural_category)
```

We create a function to generate bidimensional plots of each numeric variable vs each distance variable (bidimensional_plots_num). The function takes the artifacts_all table, the distance columns names, the variable to compare with, one label of the filtering combination and one name of a structural category.

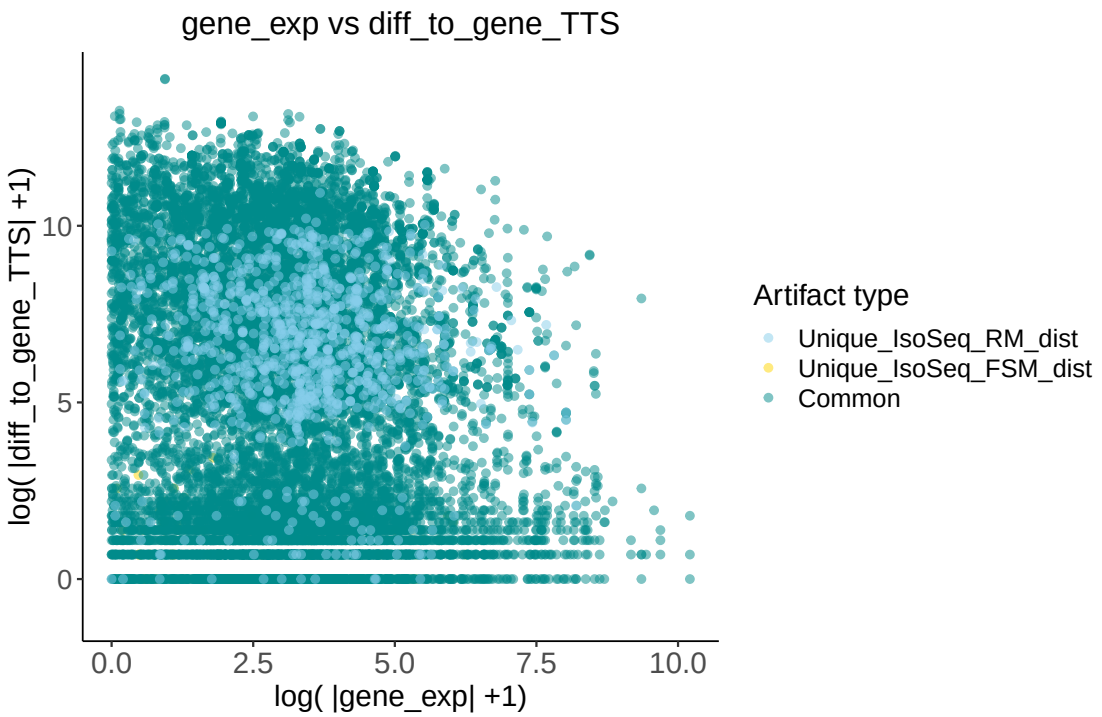
```
bidimensional_plotsnum <- function(classification,
                                   col, var,labl, sc){
  var_df <- classification %>%
    dplyr::select(structural_category,
                  source_transcriptome,
                  artifact_type, artifact_lab,
                  dplyr::all_of(col),
                  dplyr::all_of(var))

  # Rename variable column to handle during plotting
  var_df <- var_df %>% dplyr::rename(variable = var)
  # Rename distance column to handle during plotting
  var_df <- var_df %>% dplyr::rename(distance = col)

  var_df <- var_df %>%
    dplyr::filter(!(is.na(variable))& !(is.na(distance)))


  difvar<-var_df %>%
    dplyr::filter(!(source_transcriptome == labl &
                      artifact_lab == 'Common'))%>%
    dplyr::filter(structural_category==sc) %>%
    dplyr::mutate(variable = factor(variable)) %>%
    group_by(source_transcriptome,
             structural_category, artifact_lab, variable, distance) %>%
    summarize(category_count = n()) %>%
    group_by(source_transcriptome, structural_category, artifact_lab) %>%
    mutate(suma=cumsum(category_count)) %>%
    arrange(structural_category)


  difvar$variable<-as.numeric(as.character(difvar$variable))


  p<- ggplot(difvar) +
    ggtitle(paste0(var," vs ",col))+
    geom_point(aes(x = log(abs(variable)+1) , y = log(abs(distance)+1),
                   colour = factor(artifact_lab)), alpha = 0.5)+
```

```
    ylim(-1,NA)+
    labs(x=paste0("log( |",var,"| +1)"), y = paste0("log( |",col,"| +1)"))+
    scale_color_manual(values=filt_palette_un[3:5],name="Artifact type")

}
```

So that, a code like the following one will store a set of bidimensional plots for each variable desired, in this example, 'gene_exp' for each category, in this example, ISM.

```
plots_expISM<- purrr::map(distcol,
                          ~bidimensional_plotsnum(artifacts_all, .,'gene_exp', label2,'NNC'))
```

As an example, it is shown one of the plots generated:



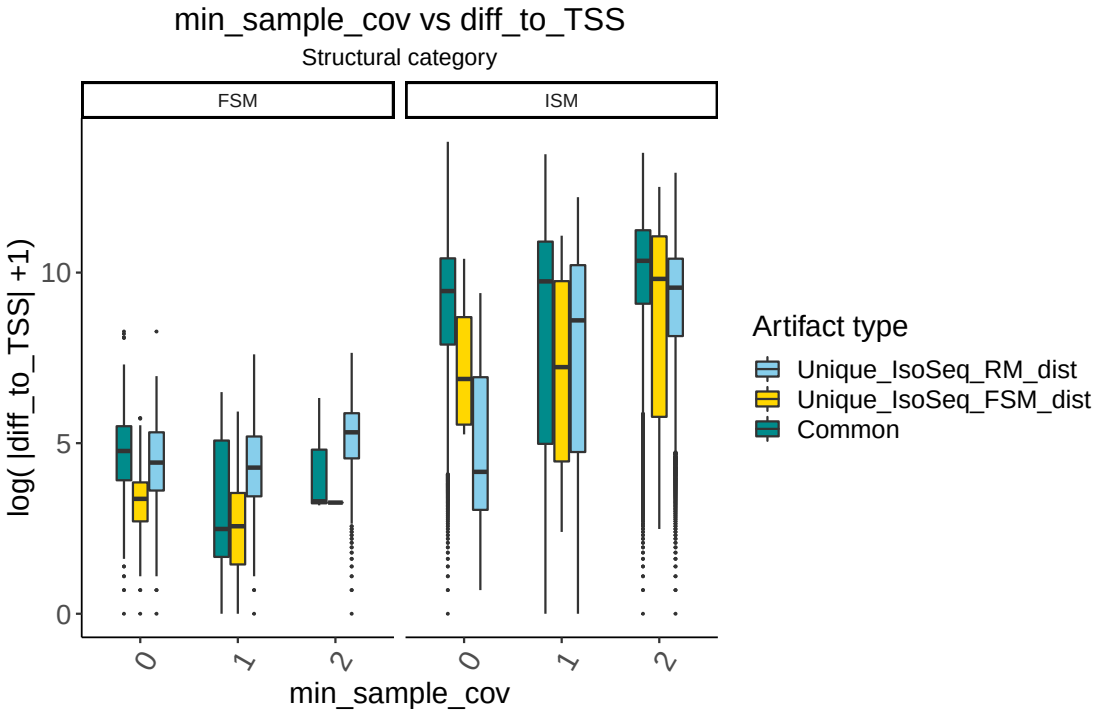
Then, we create another function to plot the rest of variables (non numeric) the same way (bidimensional_plots).

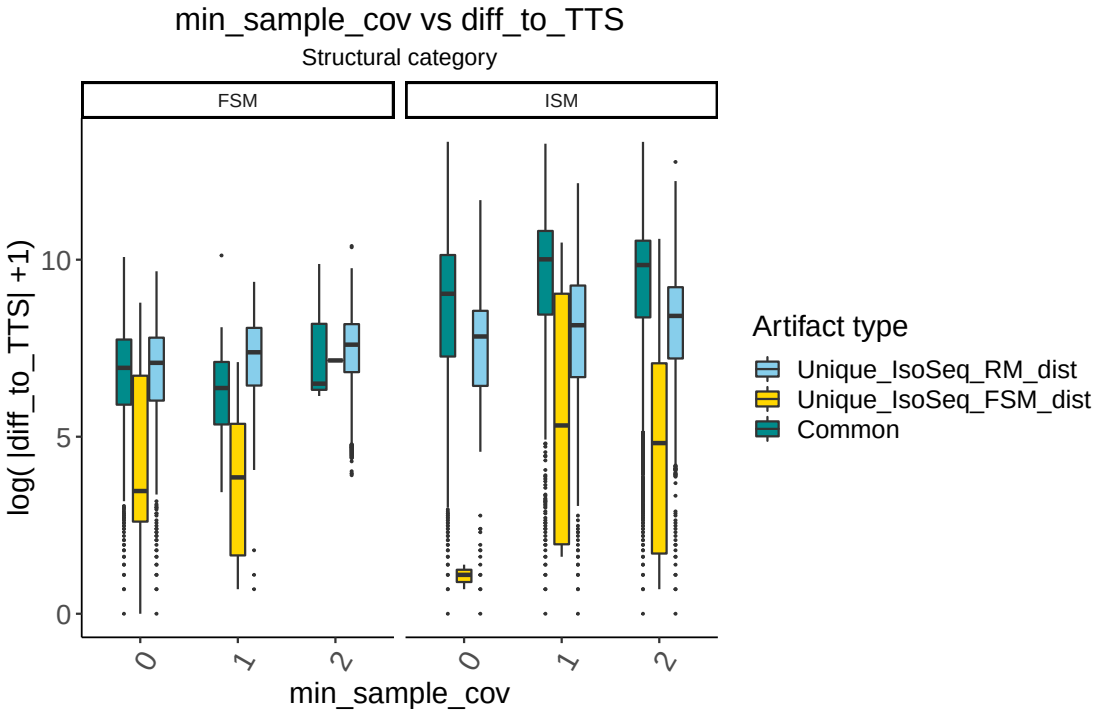
```
bidimensional_plots <- function(classification,
                                col, var,labl){
  var_df <- classification %>%
    dplyr::select(structural_category,
                  source_transcriptome,
                  artifact_type, artifact_lab,
```

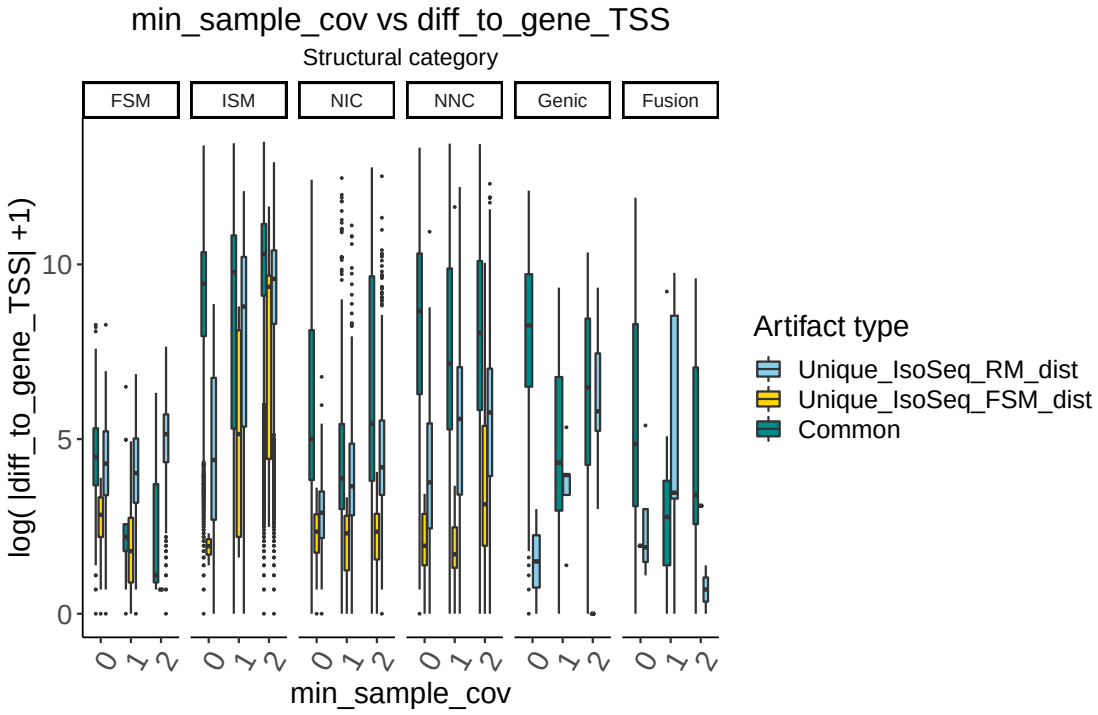
```
                  dplyr::all_of(col),
                  dplyr::all_of(var))

  # Rename variable column to handle during plotting
  var_df <- var_df %>% dplyr::rename(variable = var)
  # Rename distance column to handle during plotting
  var_df <- var_df %>% dplyr::rename(distance = col)

  var_df <- var_df %>%
    dplyr::filter(!(is.na(variable))& !(is.na(distance)))

  #plot


  diffvar<-var_df %>%
    dplyr::filter(!(source_transcriptome == labl &
                      artifact_lab == 'Common'))%>%
    dplyr::mutate(variable = factor(variable)) %>%
    group_by(source_transcriptome,
             structural_category, artifact_lab, variable, distance) %>%
    summarize(category_count = n()) %>%
    group_by(source_transcriptome, structural_category, artifact_lab) %>%
    mutate(suma=cumsum(category_count)) %>%
    arrange(structural_category)

  p<-ggplot(diffvar) +
  ggtitle(paste0(var," vs ",col))+
  geom_boxplot(aes(x = variable, y = log(abs(distance)+1),
                   fill = artifact_lab),
                outlier.size = 0.2, width = 0.5) +
    labs(subtitle= "Structural category", x=var,
         y = paste0("log( |",col,"| +1)")) +
    facet_grid(~structural_category, scales = "free")+
    scale_fill_manual(values=filt_palette_un[3:5],name="Artifact type") +
    theme(axis.text.x = element_text(angle = 60, hjust = 1))
}
```

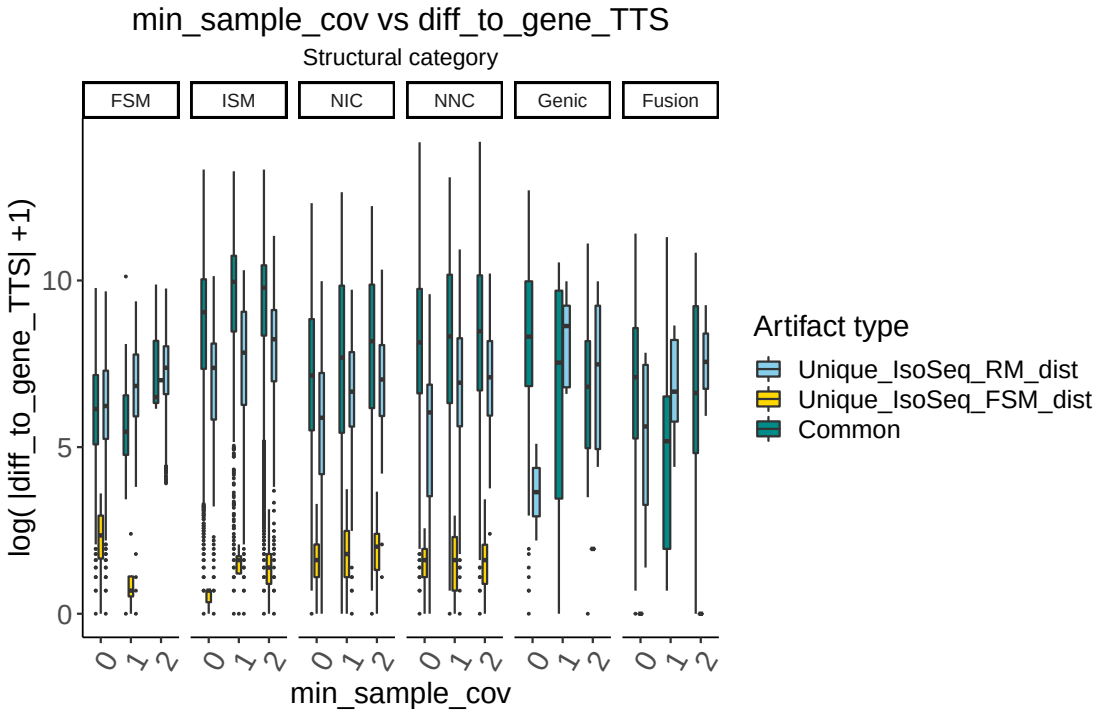
With this code we can obtain all the bidemensional plots of 'min_sample_cov' vs each distance variable:

```
plots_min<- purrr::map(distcol,
                       ~bidimensional_plots(artifacts_all, .,
                                            'min_sample_cov', label2))
```


min_sample_cov vs diff_to_TSS
Structural category
FSM
ISM
log(|diff_to_TSS| +1)
10
5
0
0
1
2
0
1
2
min_sample_cov
Artifact type
Unique_IsoSeq_RM_dist
Unique_IsoSeq_FSM_dist
Common


min_sample_cov vs diff_to_TTS
Structural category
FSM
ISM
log(|diff_to_TTS| +1)
10
5
0
0
1
2
0
1
2
min_sample_cov
Artifact type
Unique_IsoSeq_RM_dist
Unique_IsoSeq_FSM_dist
Common


min_sample_cov vs diff_to_gene_TSS
Structural category
FSM
ISM
NIC
NNC
Genic
Fusion
log(|diff_to_gene_TSS| +1)
10
5
0
0
1
2
min_sample_cov
Artifact type
Unique_IsoSeq_RM_dist
Unique_IsoSeq_FSM_dist
Common


min_sample_cov vs diff_to_gene_TTS
Structural category
FSM
ISM
NIC
NNC
Genic
Fusion
log(|diff_to_gene_TTS| +1)
10
5
0
0
1
2
min_sample_cov
Artifact type
Unique_IsoSeq_RM_dist
Unique_IsoSeq_FSM_dist
Common