Report ISW2

Alessandra Fanfano - 0282370

Il seguente report si pone l'obiettivo di descrivere il processo di analisi e studio effettuato su progetti open source di Apache, utilizzando strumenti come lo Statistical Control Process, Versioning Control Systems, Ticket Tracking e Machine Learning per fornire risultati quantitativi e statisticamente significativi.

Lo studio è stato diviso in due deliverables principali, le quali verranno di seguito analizzate in maniera separata.

Il codice sorgente delle applicazioni Java è presente nella repository :
https://github.com/Alefanfi?tab=repositories.

Le applicazioni sono quindi presenti sulla piattaforma GitHub e connesse a Travis CI, per il build automatico mediante Ant, e a SonarCloud per l'analisi del codice riguardante principalmente il controllo della presenza di code smells, vulnerabilità, duplication code ed eccessi di Cyclomatic Complexity.

Deliverable 1.

Introduzione

Per software analytics s'intende quell'insieme di metodologie, tecniche e risorse utilizzate per ottener informazioni su un sistema software target, al fine di migliorare la qualità, ottimizzare il processo produttivo oppure per trarre conclusioni su eventuali decisioni future da prendere.

La deliverable 1 ha come scopo quello di misurare la stabilità di un attributo di progetto. In particolare si è analizzato lo storico dei ticket di una specifica categoria, ad esempio "Bug Fix", "New Features" o "Fixed Ticket", di progetti open-source di Apache Software Foundation.
La stabilità di un attributo di progetto è essenziale nel momento in cui si vuole monitorare, controllare e predire il comportamento del progetto stesso durante il processo di sviluppo. La stabilità di un progetto permette di migliorare aspetti detto sviluppo e fornire quindi un prodotto di qualità maggiore. Mentre l'attività di misurazione permette di prendere decisioni basate su evidenze oggettive, assicurando una certa qualità del processo e del progetto che si sviluppa.

L'analisi dei ticket è stata effettuata tramite l'elaborazione di un grafico di tipo Process Control Chart, mettendo in relazione il numero di ticket fixed con la deviazione standard al fine di evidenziare valori che ricadono al di fuori dei valori limite di stabilità.

In questo report verranno analizzati i ticket di tipo "Fixed Ticket" relativi al progetto FALCON.

Progettazione.

È stato realizzato un programma Java per collezionare i ticket necessari all'analisi. La sua esecuzione è caratterizzata da tre fasi principali:

1. Utilizzando una API REST, fornita dalla piattaforma Jira, si collezionano tutti i ticket, relativi a bug, new features e task;
2. Il programma ottiene tutti i commit relativi alla repository di GitHub del progetto da analizzare, utilizzando le API REST fornite da GitHub stesso;
3. Il programma analizza tutti i commit, ne ricava i ticket contenuti nei messaggi. Individuati i ticket ad ognuno di essi viene associato il commit. Si prosegue in questo modo fino alla fine della lista dei commit ottenuta dalle api di GitHub.

Se nell'associare i commit ai ticket se ne individuano più di uno, che fa riferimento ad uno stesso ticket, il programma considera la data del commit più recente come data di chiusura.

Dopo aver collezionato i ticket, il programma li salva in un apposito file formato csv, riportando su due colonne distinte il ticket e la sua data, espressa in mese/anno.
Il programma, inoltre, è stato progettato in modo tale da poter analizzare diversi progetti, grazie all'utilizzo di un file di configurazione (conf.properties).

Risultati.

Il progetto "FALCON" ha un totale di 1258 ticket relativi a bug, new features e task che hanno uno stato "closed" o "resolved".

Le statistiche che sono state calcolare sono:

- Media dei ticket fixed nell'arco di tempo considerato
- Deviazione standard usata per il calcolo di Upper e Lower limit
- Numero di ticket fixed per mese

I ticket analizzati ricoprono un periodo che va da novembre 2011 fino a agosto 2018. Il numero medio di ticket chiusi al mese è pari a 15,341463. Per il calcolo del valore di Upper Limit e Lower Limit sono state utilizzate le seguenti formule:

$$\text{UpperLimit} = \text{MEAN} + 3*\text{STDDEV} = 68{,}13$$
$$\text{LowerLimiti} = \text{MEAN} - 3*\text{STDDEV} = -37{,}45$$

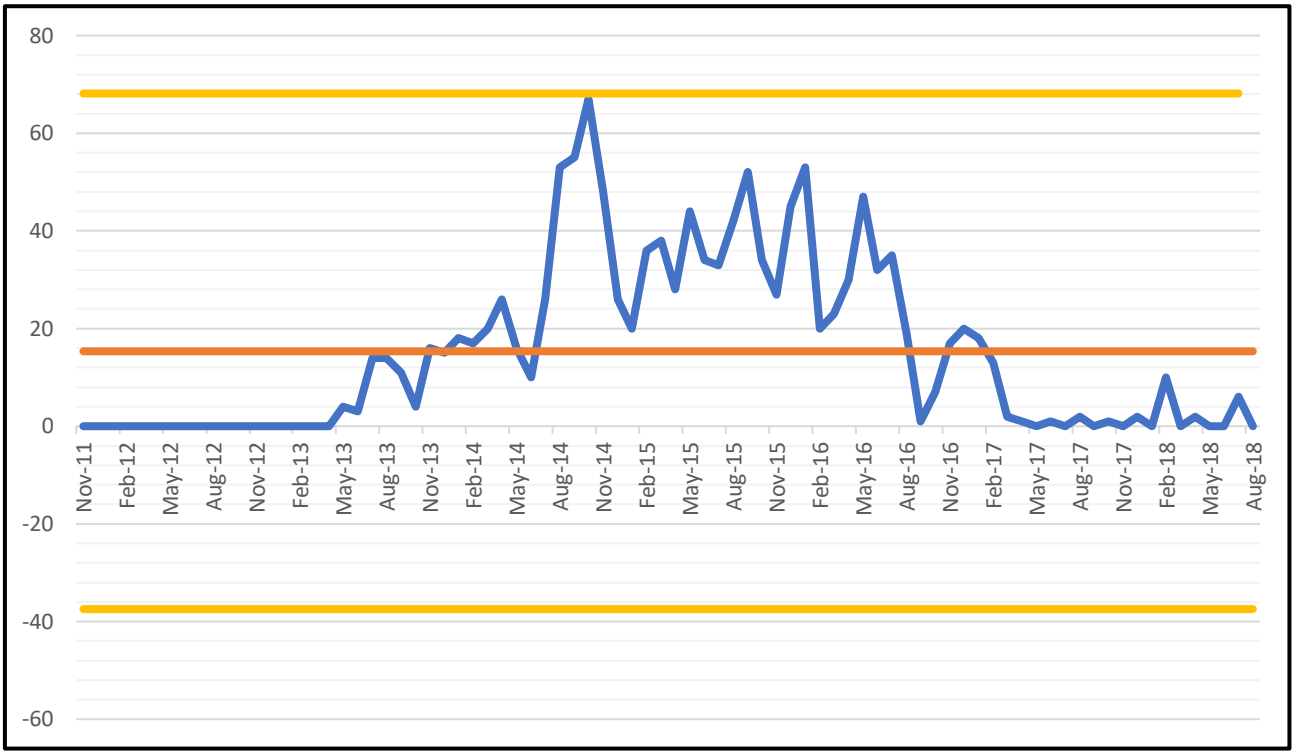

Figura 1. Process Control Chart dei tickets fixed, usando il progetto FALCON

Il Process Control Chart nel complesso risulta essere stabile: il numeri di ticket chiusi al mese, infatti non oltrepassa mai i limiti di stabilità (UpperLimit e LowerLimit). Esiste solamente un punto in cui il numero di ticket fixed, pari a 67, si avvicina molto all'upper limit individuato. Tale anomalia può essere utilizzata per individuare problemi nel processo software al fine di risolverli così da evitare che il progetto rischi di diventare instabile. In particolare la causa di tale eccezione va ricerca nel periodo antecedente.

Conclusioni.

Grazie all'utilizzo del Process Control Chart, è possibile affermare che nel complesso il progetto open-source FALCON di Apache Software Foundation risulta essere stabile per quanto riguarda l'attributo

preso in esame. È possibile tuttavia eseguire un'analisi più approfondita per individuare le cause che hanno portato ai picchi maggiori, in particolare quello verificatosi nell'ottobre del 2014. Inoltre risulta evidente come il processo di sviluppo si sia stabilizzato nel tempo, infatti, nel periodo successivo al novembre 2016 il numero di ticket fixed risulti essere vicino al valore medio per poi andare ben al di sotto di questo dal maggio del 2017.

Link.

- GitHub: https://github.com/Alefanfi/ISW2
- SonarCloud: https://sonarcloud.io/dashboard?id=alefanfi_ISW2

Deliverable 2.

La seconda deliverable ha come scopo quello di eseguire uno studio empirico, basato su modelli di Machine Learning. Grazie a tale strumento è possibile identificare le *Classi defective* di un progetto, utilizzando tecniche di *Sampling* e *Feature Selection* offerte dall'API Weka e le informazioni riguardanti le metriche dei file che compongono il progetto stesso, al fine di individuare quale combinazione di tecnica sampling/tecnica feature selection fornisce risultati migliori per ogni classificatore.

Le motivazioni che per cui si vuole individuare tali classi sono molteplici: un esempio è un'ottimizzazione delle risorse impiegate nelle attività di testing, riuscendo ad evidenziare quali sono i file devono essere soggetti a test più intensivi rispetto ad altri, basandosi quindi su dati ed evidenze oggettive.

I progetti presi in esame per lo sviluppo di tale studio empirico sono BookKeeper e AVRO.

Progettazione.

La deliverable è stata realizzata implementando un programma Java, che fosse in grado di:

- Acquisire le metriche e la defectiveness dei file java dei singoli progetti mediante l'utilizzo delle API REST di GitHub e JIRA, fornendo quindi un dataset contenente tutte le informazioni per eseguire il training e il testing dei modelli predittivi.
- Analizzare, mediante l'utilizzo di tecniche di machine learning, i dati ottenuti ed elaborati all'interno di un dataset. Viene quindi realizzato un training e testing dei classificatori, al fine di valutarne l'accuratezza.

Il programma, inoltre, è stato progettato in modo tale da poter analizzare diversi progetti, grazie all'utilizzo di un file di configurazione (conf.properties).

Generazione del dataset.

È stato realizzato un programma che permette di acquisire tutte quelle informazioni ritenute importati al fine di determinare la defectiness di un determinato file.

Per l'obiettivo posto in questo studio ha senso andare a considerare solamente i file contenenti codice java, scartando i restanti file di altri linguaggi di programmazione. Inoltre, sono state escluse il 50% delle release, quelle aventi una release date più vecchia, in quanto, prima di poter definire un file buggy, è necessario accorgersi della presenza del bug stesso, ma ciò può avvenire anche in tempi non trascurabili, class Snoring, portando così all'insicurezza sulla defectiness di un file. L'analisi di release meno recenti non rappresenta, tuttavia, la certezza che una specifica classe sia defective o meno, ma risulta comunque meno probabile che un bug non sia stato scoperto, riuscendo così ad ottenere risultati più accurati. Considerando solamente la prima metà delle versioni, infatti, la probabilità di avere delle classi "dormienti" scende al di sotto del 10%.

Il file di output che si ottiene da questa prima parte del programma fornisce un dataset per il progetto considerato, costituito dall'insieme di metriche e l'eventuale defectiness per ogni file di ogni release.

La generazione del dataset è suddivisa in due fasi:

- Raccolta dei dati necessari
- Elaborazione dei dati ottenuti nella prima fase

I dati riguardanti le release di un determinato progetto sono stati raccolti utilizzando le API REST di JIRA, più precisamente è stata fatta un'interrogazione mediante l'url seguente:

"https://issues.apache.org/jira/rest/api/2/project/"+ projName

Le release che sono state prese in considerazione sono tutte quelle che avevano una release date e per ciascuna sono state prese le informazioni

- id,
- data di rilascio della release
- nome.

Queste informazioni, una volta ottenute, sono state riordinate cronologicamente ed epurate della metà più recente, in modo tale da andare a reperire tutti gli altri dati, necessari al calcolo delle metriche, solo per quei ticket, commit e file committati rilevanti per lo studio che si intende effettuare.

Prima di procedere con il calcolo delle metriche è necessario avere la lista dei file su cui calcolarle.
Per fare ciò, si è provveduto ad individuare i ticket di tipo bug, aventi stato "closed" o "resolved" presenti all'interno del progetto e, sfruttando le informazioni già raccolte per le release, scartare tutti quelli aventi una data troppo recente. Ciascun ticket ha un proprio codice identificativo, grazie a questo è stato possibile identificare tutti quei commit che li contenevano, riuscendo così ad individuare quelle classi java "difettose". Inoltre, questi ticket, forniscono teoricamente le informazioni riguardanti le versioni affette da bug (Affected Versions – Av) e le versioni fix.
Il programma è stato quindi strutturato in modo tale da reperire tali dati, se presenti, attraverso le API REST di JIRA.

Spesso accade però che i ticket non abbiano le informazioni relative alle versioni coinvolte (AV), per questo motivo tali informazioni devono essere ricavate utilizzando la tecnica "Proportion". L'idea alla sua base è l'esistenza di una relazione di proporzionalità nella distanza che intercorre tra la Opening Version/Injected Version e quella tra la Fixed Version/Injected Version. Tale proporzionalità dovrebbe essere simile tra tutti i ticket di un medesimo progetto, quindi si calcola il valore di proporzionalità dei ticket completi di tutte le informazioni, così da poter determinare le informazioni mancanti nei restanti ticket.
Il calcolo del valore proporzionale si può effettuare utilizzando tre valori associati ai ticket:

- Opening Version (OV): la versione in cui è stato rilevato il bug. Ha come data di rilascio quella subito successiva alla data di apertura di un ticket;
- Fixed Version (FV): la versione in cui il bug è stato risolto;
- Injected Version (IV): la versione in cui si è formato il bug.

L'Opening Version si individua tramite un'interpolazione tra la data di apertura di un ticket e la release date delle versioni successive del progetto. La Fixed Version si individua tramite un'interpolazione tra la data presente nel commit che risolve il bug evidenziato dal ticket e le date delle versioni successive del progetto. Può capitare però di avere più commit associati ad uno stesso ticket, per questo motivo è bene considerare quello avente la data più recente per indicare la chiusura del ticket. L'Injected Version, invece, non è un'informazione direttamente contenuta dal ticket. Questi, infatti, possiedono una lista di tutte le versioni che sono state affette da bug. È possibile quindi identificare come Injected Version la più piccola tra quelle affette, sempre se queste sono presenti come informazione del ticket.
Il valore proporzionale P può essere calcolato tramite la seguente formula:

$$P = \frac{FV - IV}{FV - OP}$$

Per i ticket ce non forniscono l'Injected Version, si predice tale valore utilizzando la media dei valori di P dell'1% dei ticket precedenti, mettendo in pratica il calcolo di proportion con la tecnica della moving window, potendo eseguire così una stima più accurata. La formula utilizzata per il calcolo è la seguente:

$$Predicted\ IV = FV - (FV - OV) * P$$

Dopo aver valutato la diffettosità delle classi, il programma esegue la computazione delle misurazioni su di esse.
Sono state selezionate 9 metriche:

- Size : numero di linee del codice (LOC);
- Loc_Touched : numero di linee di codice aggiunte o eliminate.
- Loc_Added : numero di linee di codice aggiunte in una revisione;
- Max_Loc_Added : il massimo numero di linee di codice aggiunte in una revisione;
- Avg_Loc_Added : la media del numero di linee di codice aggiunte in una revisione;
- Nauth : il numero di autori che hanno modificato una determinata classe;
- NFix : il numero di bug fix;
- NR: numero di revisioni in cui è stata modificata la classe;
- ChgSetSize : numero di file comittati insieme.

La misurazione delle classi avviene usando il codice sorgente presente su GitHub. Inoltre, per reperire le informazioni necessarie al calcolo di queste metriche si è fatto ampio uso delle api rest fornite da tale servizio di hosting, le quali permettono di ricavare molti dati necessari. Oltre le informazioni riguardanti le release e ticket, è stato infatti possibile informazioni quali:

- Numero di linee di codice aggiunte, modifica o cancellate per ogni classe;
- L'autore di un determinato commit;
- Il contenuto di un determinato file, utilizzato principalmente per riuscire a calcolare la sua size senza considerare le linee di codice commentate;
- La data di ogni commit considerato.

Il programma è stato strutturato in modo tale da andare a reperire tutte le informazioni necessarie e, solo dopo averle tutte quante, andare ad effettuare il calcolo delle metriche e l'effettiva scrittura del dataset su un apposito file csv denominato come projName + "DatasetInfo.csv".

Il file che viene ad essere creato sarà quindi costituito da dodici colonne, oltre a quelle riportanti il valore delle metriche avremo:

- La release di quello specifico file;
- Il nome del file stesso;
- Buggy, ovvero un'indicazione sulla bugginess della classe, "Si" oppure "No".

Release	File	Size	LocTouched	LocAdded	MaxLocAdde	AvgLocAdde	Nauth	Nfix	Nr	ChgSetSize	Buggy
1	src/java/org/apache/avro/AvroRuntimeException.java	8	72	48	24	16	1	0	3	154	No
1	src/java/org/apache/avro/AvroTypeException.java	8	87	56	28	14	1	0	4	178	No
1	src/java/org/apache/avro/Protocol.java	309	1115	727	300	55.92	1	0	13	244	Si
1	src/java/org/apache/avro/Schema.java	691	2101	1413	451	88.31	1	0	16	249	Si
1	src/java/org/apache/avro/SchemaParseException.java	8	72	48	24	16	1	0	3	154	No
1	src/java/org/apache/avro/file/DataFileReader.java	200	674	446	208	74.33	1	0	6	215	No
1	src/java/org/apache/avro/file/DataFileWriter.java	170	585	389	183	64.83	1	0	6	215	No
1	src/java/org/apache/avro/file/SeekableFileInput.java	15	96	64	32	21.33	1	0	3	154	No
1	src/java/org/apache/avro/file/SeekableInput.java	19	108	72	36	24	1	0	3	154	No
1	src/java/org/apache/avro/generic/GenericArray.java	19	112	74	37	18.5	1	0	4	178	No
1	src/java/org/apache/avro/generic/GenericData.java	262	819	549	250	54.9	1	0	10	249	No
1	src/java/org/apache/avro/generic/GenericDatumReader.java	445	1748	1105	265	73.67	1	0	15	266	No
1	src/java/org/apache/avro/generic/GenericDatumWriter.java	235	707	480	164	53.33	1	0	9	244	No
1	src/java/org/apache/avro/generic/GenericRecord.java	11	84	56	28	18.67	1	0	3	154	No
1	src/java/org/apache/avro/generic/GenericRequestor.java	40	201	129	57	21.5	1	0	6	216	No
1	src/java/org/apache/avro/generic/GenericResponder.java	37	188	121	53	24.2	1	0	5	192	No
1	src/java/org/apache/avro/io/DatumReader.java	19	124	81	39	16.2	1	0	5	198	No
1	src/java/org/apache/avro/io/DatumWriter.java	17	115	75	36	15	1	0	5	198	No
1	src/java/org/apache/avro/io/ValueReader.java	436	905	688	418	137.6	1	0	5	173	No
1	src/java/org/apache/avro/io/ValueWriter.java	387	710	565	383	141.25	1	0	4	171	No
1	src/java/org/apache/avro/ipc/AvroRemoteException.java	23	136	88	44	22	1	0	4	178	No
1	src/java/org/apache/avro/ipc/ByteBufferInputStream.java	69	258	172	69	28.67	1	0	6	174	No
1	src/java/org/apache/avro/ipc/ByteBufferOutputStream.java	70	263	175	87	43.75	1	0	4	171	No
1	src/java/org/apache/avro/ipc/ByteBufferValueReader.java	1	164	82	41	16.4	1	0	5	172	No
1	src/java/org/apache/avro/ipc/ByteBufferValueWriter.java	1	176	88	44	17.6	1	0	5	172	No

Figura 2: Esempio di dataset estrapolato usando AVRO

Release	File	Size	LocTouched	LocAdded	MaxLocAdde	AvgLocAdde	Nauth	Nfix	Nr	ChgSetSize	Buggy
1	bookkeeper-benchmark/src/main/java/org/apache/bookkeeper/benchmark/MySqlClient.java	121	347	242	137	60.5	2	0	4	471	No
1	bookkeeper-benchmark/src/main/java/org/apache/bookkeeper/benchmark/TestClient.java	221	436	340	252	68	2	1	5	471	Si
1	bookkeeper/src/main/java/org/apache/bookkeeper/bookie/Bookie.java	523	545	545	545	545	1	0	1	206	No
1	bookkeeper/src/main/java/org/apache/bookkeeper/bookie/BookieException.java	62	81	81	81	81	1	0	1	206	No
1	bookkeeper/src/main/java/org/apache/bookkeeper/bookie/BufferedChannel.java	141	168	168	168	168	1	0	1	206	No
1	bookkeeper/src/main/java/org/apache/bookkeeper/bookie/EntryLogger.java	462	487	487	487	487	1	0	1	206	No
1	bookkeeper/src/main/java/org/apache/bookkeeper/bookie/FileInfo.java	102	124	124	124	124	1	0	1	206	No
1	bookkeeper/src/main/java/org/apache/bookkeeper/bookie/LedgerCache.java	513	536	536	536	536	1	0	1	206	No
1	bookkeeper/src/main/java/org/apache/bookkeeper/bookie/LedgerDescriptor.java	110	133	133	133	133	1	0	1	206	No
1	bookkeeper/src/main/java/org/apache/bookkeeper/bookie/LedgerEntryPage.java	129	151	151	151	151	1	0	1	206	No
1	bookkeeper/src/main/java/org/apache/bookkeeper/bookie/MarkerFileChannel.java	125	147	147	147	147	1	0	1	206	No
1	bookkeeper/src/main/java/org/apache/bookkeeper/client/AsyncCallback.java	111	126	126	126	126	1	0	1	206	No
1	bookkeeper/src/main/java/org/apache/bookkeeper/client/BKException.java	227	249	249	249	249	1	0	1	206	No
1	bookkeeper/src/main/java/org/apache/bookkeeper/client/BookKeeper.java	380	410	410	410	410	1	0	1	206	No
1	bookkeeper/src/main/java/org/apache/bookkeeper/client/BookieWatcher.java	182	204	204	204	204	1	0	1	206	No
1	bookkeeper/src/main/java/org/apache/bookkeeper/client/CRC32DigestManager.java	34	50	50	50	50	1	0	1	206	No
1	bookkeeper/src/main/java/org/apache/bookkeeper/client/DigestManager.java	163	184	184	184	184	1	0	1	206	No
1	bookkeeper/src/main/java/org/apache/bookkeeper/client/DistributionSchedule.java	37	61	61	61	61	1	0	1	206	No
1	bookkeeper/src/main/java/org/apache/bookkeeper/client/LedgerCreateOp.java	145	167	167	167	167	1	0	1	206	No
1	bookkeeper/src/main/java/org/apache/bookkeeper/client/LedgerDeleteOp.java	58	80	80	80	80	1	0	1	206	No
1	bookkeeper/src/main/java/org/apache/bookkeeper/client/LedgerEntry.java	60	83	83	83	83	1	0	1	206	No
1	bookkeeper/src/main/java/org/apache/bookkeeper/client/LedgerHandle.java	525	547	547	547	547	1	0	1	206	No
1	bookkeeper/src/main/java/org/apache/bookkeeper/client/LedgerMetadata.java	178	198	198	198	198	1	0	1	206	No

Figura 3: Esempio di dataset estrapolato usando BookKeeper

Link.

- GitHub: https://github.com/Alefanfi/ISW2
- SonarCloud: https://sonarcloud.io/dashboard?id=alefanfi_ISW2

Training e test dei classificatori.

La seconda parte del deliverable permette di sfruttare i dati raccolti nella prima parte e valutare quale modello di Machine Learning produce stime più accurate, provando tra le possibili combinazioni di classificatori, feature selection e tecniche di balancing. Nello studio in esame è stata presa in considerazione, come evaluation tecnique, la walk forward. Tale tecnica è la scelta migliore nel caso in cui si voglia preservare l'ordine dei dati, Time-series validation.

Per applicare questa evaluation tecnique è necessario suddividere il dataset in porzioni, dove il criterio scelto per tale suddivisione è la release: le singole unità del walk forword contengono la porzione del dataset relative a una specifica versione, ovvero l'i-esimo walk contiene i file e le loro metriche che fanno riferimento alla release i-esima.

Per questo motivo, per ogni release del progetto analizzato, il programma riceve come input dei file arff che contengono le seguenti informazioni:

- @Relation: nome del progetto su cui si sta lavorando
- @Attribute: il tipo di metriche
- @Data: il valori ottenuti dal calcolo delle metriche

Il dataset precedentemente realizzato è stato memorizzato all'interno di un file csv, quindi per evitare problemi di conversione dei file, tale processo viene ad essere automatizzato all'interno del programma stesso, così da poter anche utilizzare le API Java offerte dal Software Weka.

I classificatori che vengono presi in considerazione sono:

- Random Forest
- Ibk
- Naive Bayes

Le tecniche di balancing sono utilizzate per evitare uno sbilanciamento tra il numero di istanze delle classi buggy e no buggy, evitando così di avere bassa accuratezza sulle istanze del primo tipo ed alta accuratezza sulle istanze del secondo tipo. Quelle che sono state prese in considerazione in questo studio sono:

- No sampling
- Oversampling: vengono ripetute le istanze della classe minoritaria fino a quando non avranno numero pari a quello della classe maggioritaria;

- Undersampling: vengono estratte dalla classe maggioritaria un numero di classi pari a quello della classe minoritaria;
- SMOTE: raffinamento dell'oversampling. In questo caso l'estensione della classe minoritaria, non avviene con la ripetizione delle istanze, ma generandole in modo sintetico.

Per applicare le tecniche di oversampling e undersampling viene utilizzato lo strumento Resample offerto da Weka, andando a specificare i parametri opportuni.

Le tecniche di feature selection sono tecniche che permettono di ridurre il numero degli attributi correlati, cercando così di migliorare l'apprendimento del classificatore rispetto al caso in cui si considera l'intero dataset. Per fare ciò è stato utilizzato l'evaluator CFSSubsetEval offerto da Weka, applicando un approccio del tipo backward: vengono considerati tutti gli attributi del dataset, procedendo alla rimozione di un solo attributo per volta fino ad ottenere un significativo decremento dell'accuratezza.

Lo scopo principale di tale programma è quello di provare tutte le possibili combinazioni di classificatori, feature selection e balancing. Dai risultati dei test, il programma valuta l'accuratezza del classificatore utilizzando le seguenti metriche:

- Recall
- Precision
- AUC
- Kappa

Per ciascun progetto, il programma restituisce un file csv contenente i dati dei test:

- Nome del progetto
- Numero di release considerate nel training
- Percentuale di dati utilizzati nel training
- Percentuale di classi difettose nel dataset di training
- Classificatore utilizzato
- Tecnica di balancing utilizzata
- Tecnica di feature selection (indicando se è stata utilizzata o meno)
- Numero di True Positive
- Numero di False Positive
- Numero di True Negative
- Numero di False Negative
- Precision
- Recall
- ROC Area
- Kappa

Dataset	#Training	%Training	%DefectTrai	%DefectTes	Classifier	Balancing	FeatureSele	TP	FP	TN	FN	Precision	Recall	ROC Area	Kappa
AVRO	1	0.4519774	0.1125	0.185567	IBK	NO SAMPLING	Yes	79.0	12.0	6.0	0.0	0.8681318681318682	1.0	0.6835443037974683	0.44886363636363624
AVRO	1	0.4519774	0.1125	0.185567	RandomForest	NO SAMPLING	Yes	158.0	30.0	6.0	0.0	0.8404255319148937	1.0	0.7794479606188467	0.24572317262830473
AVRO	1	0.4519774	0.1125	0.185567	NaiveBayes	NO SAMPLING	Yes	237.0	48.0	6.0	0.0	0.8315789473684211	1.0	0.7146819815596187	0.1691648822269807
AVRO	1	0.4519774	0.1125	0.185567	IBK	OVERSAMPLING	Yes	79.0	11.0	7.0	0.0	0.8777777777777778	1.0	0.769690576652602	0.5089737689829726
AVRO	1	0.4519774	0.1125	0.185567	RandomForest	OVERSAMPLING	Yes	158.0	29.0	7.0	0.0	0.8449197860962567	1.0	0.7677566807313643	0.28221485072722624
AVRO	1	0.4519774	0.1125	0.185567	NaiveBayes	OVERSAMPLING	Yes	237.0	47.0	7.0	0.0	0.8345070422535211	1.0	0.6795593061415847	0.19523389232127122
AVRO	1	0.4519774	0.1125	0.185567	IBK	UNDERSAMPLIN	Yes	78.0	13.0	5.0	1.0	0.8571428571428571	0.9873417721518988	0.6325597749648383	0.35700757575757547
AVRO	1	0.4519774	0.1125	0.185567	RandomForest	UNDERSAMPLIN	Yes	157.0	30.0	6.0	1.0	0.839572192513369	0.9936708860759493	0.7216068917018285	0.2327124266394489
AVRO	1	0.4519774	0.1125	0.185567	NaiveBayes	UNDERSAMPLIN	Yes	233.0	47.0	7.0	4.0	0.8321428571428572	0.9831223628691983	0.6365057040162525	0.1628025046539177
AVRO	1	0.4519774	0.1125	0.185567	IBK	SMOTE	Yes	79.0	11.0	7.0	0.0	0.8777777777777778	1.0	0.709915611814346	0.5089737689829726
AVRO	1	0.4519774	0.1125	0.185567	RandomForest	SMOTE	Yes	158.0	29.0	7.0	0.0	0.8449197860962567	1.0	0.7834036568213784	0.28221485072722624
AVRO	1	0.4519774	0.1125	0.185567	NaiveBayes	SMOTE	Yes	237.0	37.0	17.0	0.0	0.864963503649635	1.0	0.8112986404125645	0.42804780876494025
AVRO	1	0.4519774	0.1125	0.185567	IBK	NO SAMPLING	No	79.0	12.0	6.0	0.0	0.8681318681318682	1.0	0.6835443037974683	0.44886363636363624
AVRO	1	0.4519774	0.1125	0.185567	RandomForest	NO SAMPLING	No	158.0	30.0	6.0	0.0	0.8404255319148937	1.0	0.7794479606188467	0.24572317262830473
AVRO	1	0.4519774	0.1125	0.185567	NaiveBayes	NO SAMPLING	No	237.0	48.0	6.0	0.0	0.8315789473684211	1.0	0.7146819815596187	0.1691648822269807
AVRO	1	0.4519774	0.1125	0.185567	IBK	OVERSAMPLING	No	79.0	11.0	7.0	0.0	0.8777777777777778	1.0	0.769690576652602	0.5089737689829726
AVRO	1	0.4519774	0.1125	0.185567	RandomForest	OVERSAMPLING	No	158.0	29.0	7.0	0.0	0.8449197860962567	1.0	0.7677566807313643	0.28221485072722624
AVRO	1	0.4519774	0.1125	0.185567	NaiveBayes	OVERSAMPLING	No	237.0	47.0	7.0	0.0	0.8345070422535211	1.0	0.6795593061415847	0.19523389232127122
AVRO	1	0.4519774	0.1125	0.185567	IBK	UNDERSAMPLIN	No	78.0	13.0	5.0	1.0	0.8571428571428571	0.9873417721518988	0.6325597749648383	0.35700757575757547
AVRO	1	0.4519774	0.1125	0.185567	RandomForest	UNDERSAMPLIN	No	157.0	30.0	6.0	1.0	0.839572192513369	0.9936708860759493	0.7216068917018285	0.2327124266394489
AVRO	1	0.4519774	0.1125	0.185567	NaiveBayes	UNDERSAMPLIN	No	233.0	47.0	7.0	4.0	0.8321428571428572	0.9831223628691983	0.6365057040162525	0.1628025046539177

Figura 4: Esempio file csv estrapolato da AVRO

Dataset	#Training	%Training	%DefectTrai	%DefectTes	Classifier	Balancing	FeatureSele	TP	FP	TN	FN	Precision	Recall	ROC Area	Kappa
BOOKKEEPER	1	0.61178863	0.33222592	0.6806283	IBK	NO SAMPLING	Yes	46.0	13.0	117.0	15.0	0.7796610169491526	0.7540983606557377	0.8241488020176545	0.6598397150489759
BOOKKEEPER	1	0.61178863	0.33222592	0.6806283	RandomForest	NO SAMPLING	Yes	89.0	20.0	240.0	33.0	0.8165137614678899	0.7295081967213115	0.862468474148802	0.6715764202056905
BOOKKEEPER	1	0.61178863	0.33222592	0.6806283	NaiveBayes	NO SAMPLING	Yes	91.0	23.0	367.0	92.0	0.7982456140350878	0.4972677595628415	0.890444164214656	0.4870267871739182
BOOKKEEPER	1	0.61178863	0.33222592	0.6806283	IBK	OVERSAMPLING	Yes	50.0	29.0	101.0	11.0	0.632911392405063	0.819672131147541	0.8154476670870113	0.5532686235528008
BOOKKEEPER	1	0.61178863	0.33222592	0.6806283	RandomForest	OVERSAMPLING	Yes	87.0	35.0	225.0	35.0	0.7131147540983607	0.7131147540983607	0.846374527112232	0.578499369482976
BOOKKEEPER	1	0.61178863	0.33222592	0.6806283	NaiveBayes	OVERSAMPLING	Yes	144.0	50.0	340.0	39.0	0.7422680412371134	0.7868852459016393	0.8702466022138153	0.6483377810877347
BOOKKEEPER	1	0.61178863	0.33222592	0.6806283	IBK	UNDERSAMPLING	Yes	43.0	24.0	106.0	18.0	0.6417910447761194	0.7049180327868853	0.758953341740227	0.5070664864200566
BOOKKEEPER	1	0.61178863	0.33222592	0.6806283	RandomForest	UNDERSAMPLING	Yes	67.0	26.0	234.0	55.0	0.7204301075268817	0.5491803278688525	0.8109867591424968	0.4794239375483697
BOOKKEEPER	1	0.61178863	0.33222592	0.6806283	NaiveBayes	UNDERSAMPLING	Yes	124.0	42.0	348.0	59.0	0.7469879518072289	0.6775956284153005	0.8496006725514922	0.5843084017497361
BOOKKEEPER	1	0.61178863	0.33222592	0.6806283	IBK	SMOTE	Yes	44.0	12.0	118.0	17.0	0.7857142857142857	0.7213114754098361	0.8117276166456494	0.6429906542056075
BOOKKEEPER	1	0.61178863	0.33222592	0.6806283	RandomForest	SMOTE	Yes	76.0	15.0	245.0	46.0	0.8351648351648352	0.6229508196721312	0.8618379571248423	0.606132314661438
BOOKKEEPER	1	0.61178863	0.33222592	0.6806283	NaiveBayes	SMOTE	Yes	117.0	29.0	361.0	66.0	0.8013698630136986	0.639344262295082	0.9003082527672691	0.5970195660381549
BOOKKEEPER	1	0.61178863	0.33222592	0.6806283	IBK	NO SAMPLING	No	46.0	13.0	117.0	15.0	0.7796610169491526	0.7540983606557377	0.8241488020176545	0.6598397150489759
BOOKKEEPER	1	0.61178863	0.33222592	0.6806283	RandomForest	NO SAMPLING	No	89.0	20.0	240.0	33.0	0.8165137614678899	0.7295081967213115	0.862468474148802	0.6715764202056905
BOOKKEEPER	1	0.61178863	0.33222592	0.6806283	NaiveBayes	NO SAMPLING	No	91.0	23.0	367.0	92.0	0.7982456140350878	0.4972677595628415	0.890444164214656	0.4870267871739182
BOOKKEEPER	1	0.61178863	0.33222592	0.6806283	IBK	OVERSAMPLING	No	50.0	29.0	101.0	11.0	0.632911392405063	0.819672131147541	0.8154476670870113	0.5532686235528008
BOOKKEEPER	1	0.61178863	0.33222592	0.6806283	RandomForest	OVERSAMPLING	No	87.0	35.0	225.0	35.0	0.7131147540983607	0.7131147540983607	0.846374527112232	0.578499369482976
BOOKKEEPER	1	0.61178863	0.33222592	0.6806283	NaiveBayes	OVERSAMPLING	No	144.0	50.0	340.0	39.0	0.7422680412371134	0.7868852459016393	0.8702466022138153	0.6483377810877347
BOOKKEEPER	1	0.61178863	0.33222592	0.6806283	IBK	UNDERSAMPLING	No	43.0	24.0	106.0	18.0	0.6417910447761194	0.7049180327868853	0.758953341740227	0.5070664864200566
BOOKKEEPER	1	0.61178863	0.33222592	0.6806283	RandomForest	UNDERSAMPLING	No	67.0	26.0	234.0	55.0	0.7204301075268817	0.5491803278688525	0.8109867591424968	0.4794239375483697

Figura 5: esempio file csv estrapolato da BookKeeper

Risultati.

L'analisi dei risultati si concentra sul valutare inizialmente l'efficacia delle diverse combinazioni delle tecniche precedentemente citate.

Nel seguito del documento si parlerà di distribuzioni/aree simmetriche in riferimento al valore mediano (o mediana) del singolo box plot, riportato in figura.

Per riuscire a illustrare i risultati ottenuti si è ritenuta una buona scelta quella di utilizzare il software di JMP, creando uno script che genera i grafici che verranno presentati di seguito.

AVRO.

Sampling e Feature Selection

Figura 6: Accuratezza dei classificatori al variare delle tecniche di feature selection e sampling

Per il classificatore Ibk si osserva come ogni metrica presenta una maggiore dispersione dei dati verso il valore minimo, infatti sono presenti anche valori di outliers. Più nel dettaglio:

- ***Precision***: nel caso la feature selection non sia applicata, si ottengono buoni risultati con la tecnica di SMOTE in quanto il diagramma risulta essere relativamente simmetrico e di area

piccola. Nel caso in cui si utilizzi la feature selection si ottengono buoni risultati applicando la tecnica di balancing di oversampling e non applicando alcuna tecnica.

- ***Recall:*** nel caso la feature selection non sia applicata, si ottengono buoni risultati con tutte le tecniche in quanto il diagramma risulta essere relativamente simmetrico e di area piccola. Nel caso in cui si utilizzi la feature selection si ottengono risultati analoghi per la tecnica SMOTE e senza applicare alcuna tecnica di balancing. Inoltre, si riscontra una dispersione maggiore verso l'alto e verso il basso, rispettivamente per le tecniche undersampling e oversampling.
- ***ROCArea:*** nel caso la feature selection non sia applicata e nel caso in cui è applicata, si ottengono buoni risultati con la tecnica di balacing di undersampling in quanto il diagramma risulta essere relativamente simmetrico e di area piccola. Inoltre si ottengo risultati abbastanza analoghi per la tecnica di SMOTE e il caso in cui non viene utilizzata alcuna tecnica di balancing.
- ***Kappa***: nel caso la feature selection non sia applicata, si ottengono buoni risultati con la tecnica di oversampling in quanto il diagramma risulta essere relativamente simmetrico e di area piccola. Mentre si hanno maggiori dispersioni con la tecnica di SMOTE e senza l'applicazione di tecniche di balancing, invece la tecnica di undersampling risulta essere abbastanza simmetrica, ma avente una maggiore dispersione. Nel caso in cui si utilizzi la feature selection, si osservano dispersioni simili, per il valore massimo e il valore minimo, per tutte le tecniche, si ottengono risultati discreti applicando la tecnica di balancing di undersampling, in quanto l'area del grafico risulta essere più piccola e più simmetrica.

Per il classificatore Naive Bayes si osserva come ogni metrica presenta una maggiore dispersione dei dati verso il valore minimo, infatti sono presenti anche valori di outliers. Più nel dettaglio:

- ***Precision***: nel caso la feature selection non sia applicata, si ottengono buoni risultati indipendentemente dalla tecnica usata, in quanto il diagramma risulta essere di area relativamente piccola. I risultati ottenuti sono analoghi nel caso in cui si utilizzi la feature selection. Un minimo miglioramento viene riscontrato nella tecnica undersampling.
- ***Recall:*** nel caso la feature selection non sia applicata, si ottengono buoni risultati con tutte le tecniche in quanto il diagramma risulta essere relativamente simmetrico e di area piccola, in particolare come migliore possiamo considerare il risultato ottenuto con la tecnica oversampling. Nel caso in cui si utilizzi la feature selection si ottengono risultati analoghi per la tecnica SMOTE e oversampling. Inoltre, si riscontra una dispersione leggera con l'applicazione delle altre tecniche, verso l'alto, ovvero verso il valore massimo.
- ***ROCArea:*** nel caso la feature selection non sia applicata si ottengono risultati simili con ogni tecnica di balacing in termini di dispersione; mentre per quanto riguarda la simmetria la tecnica migliore sembrerebbe essere l'undersampling. Nel caso in cui la tecnica di feature selection sia, invece, applicata si ottengono risultati simili a coppie applicando le tecniche di SMOTE e undersampling oppure applicando la tecnica di oversampling e non applicando tecniche di balancing.
- ***Kappa***: nel caso la feature selection non sia applicata, si ottengono buoni risultati con la tecnica di undersampling. Nel caso in cui si utilizzi la feature selection, si osservano dispersioni simili, per il valore massimo e il valore minimo, per tutte le tecniche, si ottengono risultati discreti applicando la tecnica di balancing di undersampling, in quanto l'area del grafico risulta essere più piccola e più simmetrica.

Per il classificatore Random Forest si osserva come ogni metrica presenta una maggiore dispersione dei dati verso il valore minimo, infatti sono presenti anche valori di outliers. Più nel dettaglio:

- ***Precision***: nel caso la feature selection non sia applicata, si ottengono buoni risultati indipendentemente dalla tecnica usata, in quanto il diagramma risulta essere di area relativamente piccola. I risultati ottenuti sono analoghi nel caso in cui si utilizzi la feature selection. Un minimo miglioramento viene riscontrato nella tecnica undersampling.
- ***Recall:*** nel caso la feature selection non sia applicata, si ottengono buoni risultati con tutte le tecniche in quanto il diagramma risulta essere relativamente simmetrico e di area piccola, in particolare come migliore possiamo considerare il risultato ottenuto con la tecnica oversampling. Nel caso in cui si utilizzi la feature selection si ottengono risultati analoghi per la tecnica SMOTE e oversampling. Inoltre, si riscontra una dispersione leggera con l'applicazione delle altre tecniche, verso l'alto, ovvero verso il valore massimo.

- ***ROCArea:*** nel caso la feature selection non sia applicata si ottengono risultati simili con ogni tecnica di balacing in termini di dispersione; mentre per quanto riguarda la simmetria la tecnica migliore sembrerebbe essere l'undersampling. Nel caso in cui la tecnica di feature selection sia, invece, applicata si ottengono risultati simili a coppie applicando le tecniche di SMOTE e undersampling oppure applicando la tecnica di oversampling e non applicando tecniche di balancing.
- ***Kappa***: nel caso la feature selection non sia applicata, si ottengono buoni risultati con la tecnica di undersampling. Nel caso in cui si utilizzi la feature selection, si osservano dispersioni simili, per il valore massimo e il valore minimo, per tutte le tecniche, si ottengono risultati discreti applicando la tecnica di balancing di undersampling, in quanto l'area del grafico risulta essere più piccola e più simmetrica.

BookKeeper.

Sampling e Feature Selection

Figura 7: Accuratezza dei classificatori al variare delle tecniche di feature selection e sampling

Per il classificatore Ibk:

- ***Precision***: nel caso la feature selection non sia applicata, si ottengono buoni risultati con la tecnica di SMOTE in quanto il diagramma risulta essere relativamente simmetrico e di area piccola. Nel caso in cui si utilizzi la feature selection si ottengono risultati analoghi applicando le tecniche di SMOTE, oversampling e non usando alcuna tecnica.
- ***Recall:*** nel caso la feature selection non sia applicata, si ottengono buoni risultati non applicando alcuna tecnica di balacing il grafico risulta essere di area piccola e relativamente simmetrico. Nel caso in cui si utilizzi la feature selection si ottengono risultati analoghi per ogni tecnica con valore mediano prossimo al valore massimo.
- ***ROCArea:*** nel caso la feature selection non sia applicata e nel caso in cui è applicata, si ottengono buoni risultati con la tecnica di balacing di oversampling in quanto il diagramma risulta essere relativamente simmetrico e di area piccola. Inoltre, si ottengono risultati abbastanza analoghi per la tecnica di SMOTE e il caso in cui non viene utilizzata alcuna tecnica di balancing.

- ***Kappa***: nel caso la feature selection non sia applicata e nel caso in cui questa viene ad essere utilizzata si ottengono risultati similari con la tecnica di SMOTE e non applicando tecniche di balancing.

Per il classificatore Naive Bayes:

- ***Precision***: nel caso la feature selection non sia applicata, si ottengono buoni quasi identici con la tecnica di SMOTE e non applicando tecniche di balancing. Nel caso in cui si utilizzi la feature selectionsi ha un notevole miglioramento con la tecnica di undersampling, in quanto l'area è piccola e simmetrica.
- ***Recall:*** nel caso la feature selection non sia applicata, il miglior risultato si ottiene con l'utilizzo di SMOTE in quanto il diagramma risulta essere relativamente simmetrico e di area piccola. Nel caso in cui si utilizzi la feature selection si ottengono risultati simili, si evidenzia però come quelli ottenuti con la tecnica undersampling risulti avere una distribuzione più simmetrica.
- ***ROCArea:*** nel caso la feature selection non sia applicata si ottengono risultati simili con l'utilizzo di SMOTE e no sampling, anche se sembrerebbe che quest'ultimo a un'area leggermente minore e quindi una dispersione minore. Nel caso in cui la tecnica di feature selection sia, invece, applicata si ottengono risultati migliori per quanto riguarda l'area della distribuzione considerata utilizzando la tecnica di undersampling, mentre se si osserva la simmetria del grafico il risultato che colpisce è quello ottenuto applicando SMOTE.
- ***Kappa***: nel caso la feature selection non sia applicata si ottengono risultati migliori per quanto riguarda l'area della distribuzione considerata utilizzando la tecnica di undersampling, mentre se si osserva la simmetria del grafico il risultato che colpisce è quello ottenuto applicando SMOTE. Nel caso in cui la tecnica di feature selection si ottengono risultati simili per tutte le tecniche di balancing, si nota solamente una dispersione minore con la tecnica di undersampling.

Per il classificatore Random Forest:

- ***Precision***: si ottengono buoni risultati nel caso in cui la feature selection e le tecniche di balancing non sono applicate. Nel caso in cui si utilizzi la feature selection il risultato migliore si ottiene applicando la tecnica di undersampling.
- ***Recall:*** sia nel caso in cui la feature selection è apllica che quando questa non lo è il risultato migliore che si ottiene è utilizzando la tecnica di balancing dell'oversampling.
- ***ROCArea:*** nel caso la feature selection non sia applicata si ottengono risultati quasi identici con l'applicazione della tecnica di oversampling e di SMOTE. Nel caso in cui la tecnica di feature selection sia, invece, applicata si ottengono risultati migliori non applicando balancing.
- ***Kappa***: nel caso la feature selection non sia applicata si ottengono risultati migliori per quanto riguarda l'area della distribuzione considerata utilizzando la tecnica di undersampling, mentre se si osserva la simmetria del grafico il risultato che colpisce è quello ottenuto applicando SMOTE. Nel caso in cui la tecnica di feature selection si ottengono risultati simili per tutte le tecniche di balancing, si nota solamente una dispersione minore con la tecnica di undersampling.

Link.

- GitHub: https://github.com/Alefanfi/ISW2Weka
- SonarCloud: https://sonarcloud.io/dashboard?id=alefanfi_ISW2Weka