Machine Learning for Software Engineering

Purificato Marco – Matricola: 0339286

AGENDA

- Introduzione
- Obiettivi
- Metodologia
- Risultati ed Analisi
- Link Utili

Quanto costano alle aziende i BUG all’interno del loro software?

$2.08 trilioni di dollari

BUDGET

Controllo della Qualità

- FREQUENTE
- AUTOMATICO
- EFFICACE

PROGETTAZIONE ED IMPLEMENTAZIONE DI TEST

Come RIDURRE l’effort legato alla fase di testing, senza compromettere la sua EFFICACIA ed il BUDGET?

INDIVIDUARE CLASSI CHE CONTENGONO BUG TRAMITE

MACHINE LEARNING

FUNZIONA IN MANIERA AUTOMATICA ED ACCURATA

I MODELLI SONO GRATUITI

LA POTENZA COMPUTAZIONALE ODIERNA CI PERMETTE UNA STIMA EFFICIENTE

OBIETTIVI

IDENTIFICARE LE CONFIGURAZIONI PREDITTIVE OTTIMALI

APPLICARE MODELLI DI MACHINE LEARNING PER OTTENERE PREDIZIONI SULLA BUGGINESS DELLE CLASSI DEI PROGETTI APACHE: BOOKKEEPER E SYNCOPE

MOSTRARE L'ANDAMENTO DELLE PRESTAZIONI DEI MODELLI USATI, AL VARIARE DELLE METRICHE E DELLE TECNICHE APPLICATE AI DATASET DI «ADDESTRAMENTO»

DUE CONCETTI FONDAMENTALI NEL MACHINE LEARNING

TESTING SET

Set di dati misurati fino all'ultima release, utilizzati per validare il modello di predizione creato con il training set

TRAINING SET

Set di dati misurati con cui viene costruito il modello di predizione

PER LE LORO MISURAZIONI SI SONO USATI I SEGUENTI STRUMENTI:

JIRA: ISSUE TRACKING SYSTEM PER LA RACCOLTA DEI TICKET FIXATI ED ETICHETTATI «BUG», ATTRAVERSO L'UTILIZZO DELLE **API JSON**.

GIT: VERSION CONTROL SYSTEM PER LA RACCOLTA DEI COMMIT ED I FILE RIFERITI, ATTRAVERSO IL PLUGIN **JGIT**; DOVE OGNI COMMIT FA RIFERIMENTO AD UN TICKET JIRA.

PROPORTION

PROPORZIONALITÀ TRA L'INTERVALLO DI RELEASE (IV, FV) E L'INTERVALLO DI RELEASE (OV, FV)

COSTANTE DI PROPORZIONALITÀ

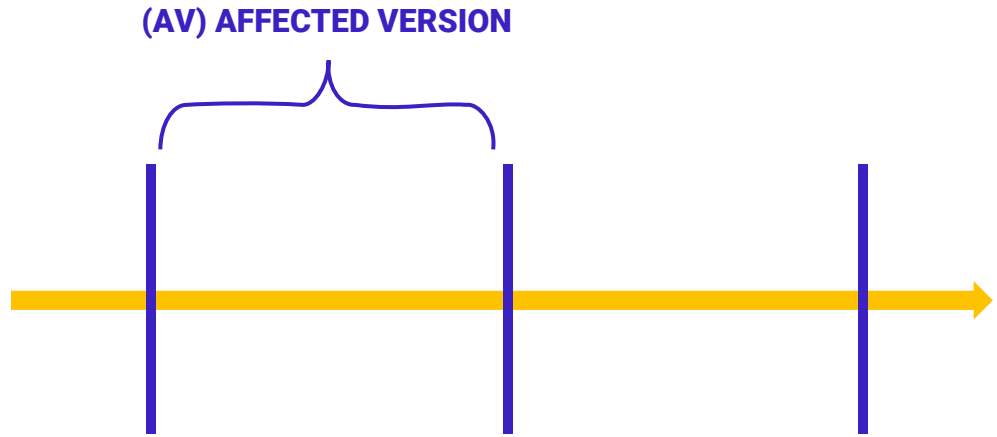
$$p = \frac{FV - IV}{FV - OV}$$

(IV) INJECTED VERSION

$$IV = FV - (FV - OV) \cdot p$$

Ogni ticket JIRA tiene traccia delle versioni relative al ciclo di vita di un difetto.

A differenza di OP ed FV che sono sempre disponibili nel ticket, indicando creazione e risoluzione del ticket stesso, non sempre troviamo indicata la injected version, necessaria al labeling delle classi.



(IV) INJECTED VERSION: RELEASE IN CUI IL BUG È STATO INTRODOTTO

(OV) OPENING VERSION: RELEASE IN CUI IL BUG È STATO RILEVATO A SEGUITO DI UNA FAILURE DEL SISTEMA

(FV) FIXED VERSION: RELEASE IN CUI IL BUG È STATO ELIMINATO

DUE VARIANTI DI PROPORTION UTILIZZATE

COLD START

Nel caso in cui i BUG con injected version disponibili fino al momento del calcolo, siano meno di una *threshold* di 5, i dati a disposizione sono considerati troppo pochi; in tal caso la costante di proporzionalità viene calcolata come mediana di quelle, riferite temporalmente prima del BUG di cui si vuole calcolare la injected version, di altri progetti della Apache Software Foundation: **TAJO, STORM, SYNCOPE, ZOOKEEPER, OPENJPA, AVRO** e **BOOKKEEPER**.

INCREMENTAL

La costante di proporzionalità è uguale alla media dei vari Proportion dei BUG aventi injected version e riferiti temporalmente prima del BUG a cui si vuole applicare Proportion per il calcolo della IV.

ASSUNZIONI

- Dato che **una funzionalità presente non è osservata come una passata**, si potrebbe incappare nel fenomeno dello *Snoring*, che consiste nell'avere una classe non buggy che in realtà ha dei difetti non emersi, *bug dormienti*; per evitare questo fenomeno si scarta la seconda metà delle release, dal momento che è molto probabile che tutti i bug presenti nella prima metà siano stati scoperti.

- Nell'analisi non si considerano i ticket che non presentano al loro interno opening version e/o fixed version e che possiedono fixed version, opening version ed injected version temporalmente incoerenti: **OV > FV** e/o **IV > OV**.

- Nell'uso di Proportion, nel caso in cui **FV = OV**, si considera **FV–OV=1**, così da evitare denominatore nullo nel calcolo della costante di proporzionalità e rimanere in un caso realistico, dove appunto non ci può essere indicato nel medesimo ticket **IV = FV**, ergo la release indicata, dove è stato introdotto il bug non può essere contemporaneamente la prima release in cui il bug è assente.

METRICHE CONSIDERATE

Il dataset ricavato dai tool di misurazione ed utilizzato per addestrare il modello di predizione, è composto dalla seguenti feature (od attributi) *intra-release*, tutte riferite alle classi:

- **LOC:** Numero di linee di codice;
- **LOC ADDED:** Somma delle LOC aggiunte;
- **LOC MAX ADDED:** LOC massime aggiunte;
- **LOC TOUCHED:** Somma delle LOC aggiunte e delle LOC rimosse;
- **NUMBER OF REVISION:** Numero di revisioni della classe;
- **AVERAGE LOC ADDED:** Media di LOC aggiunte;
- **NUMBER OF AUTHORS:** Numero di revisori della classe;
- **CHURN:** | LOC aggiunte – LOC rimosse |;
- **MAX CHURN:** CHURN massimo;
- **AVERAGE CHURN:** CHURN medio;

METRICHE DI CLASSE FORNITE

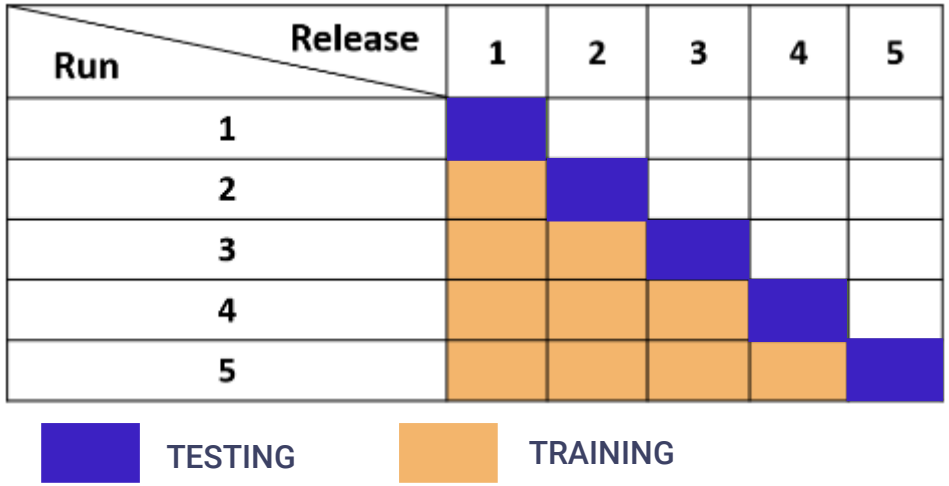
- **NUMBER OF PUBLIC METHODS (NPM):** Numero di metodi pubblici nella classe;
- **NUMBER OF PRIVATE METHODS (NPVM):** Numero di metodi privati nella classe;
- **NUMBER OF STATIC METHODS (NSM):** Numero di metodi statici nella classe;
- **NUMBER OF ALL METHODS (NAM):** Numero di metodi presenti nella classe;
- **NUMBER OF COMMENTS (NLOCM):** Numero di righe commentate;

METRICHE DI CLASSE AGGIUNTE

- **BUGGY:** Metrica binaria relativa alla bugginess della classe nella release considerata.

WALK FORWARD

VALUTAZIONE DEI CLASSIFICATORI



Per stabilire quale classificatore abbia le prestazioni migliori e con quali tecniche di utilizzo, si fa uso della tecnica di valutazione del walk forward.

IL WALK FORWARD È UNA TECNICA TIME-SERIES, CIOÈ LAVORA SU DATI SENSIBILI AL TEMPO, TENENDO CONTO QUINDI, DELL'ORDINE TEMPORALE DEI DATI: NON SI POSSONO UTILIZZARE DATI DEL FUTURO PER PREDIRRE IL PASSATO.

Il dataset è diviso in gruppi, in particolare in release **ordinate cronologicamente**, ad ogni run costruisco il training set ed il relativo labeling con le informazioni disponibili fino a quel momento, prime k release, e costruisco il testing set con le informazioni disponibili nella release k+1esima. La prima iterazione con solo il testing set è stata scartata.

CLASSIFICATORI, TECNICHE E METRICHE CONSIDERATE

CLASSIFICATORI
- Random Forest
- Naive Bayes
- IBk

COST SENSITIVE [CFN = 10 * CFP]
- Sensitive Learning
- Sensitive Threshold

FEAUTURE SELECTION: Best First - Backward e Forward Search

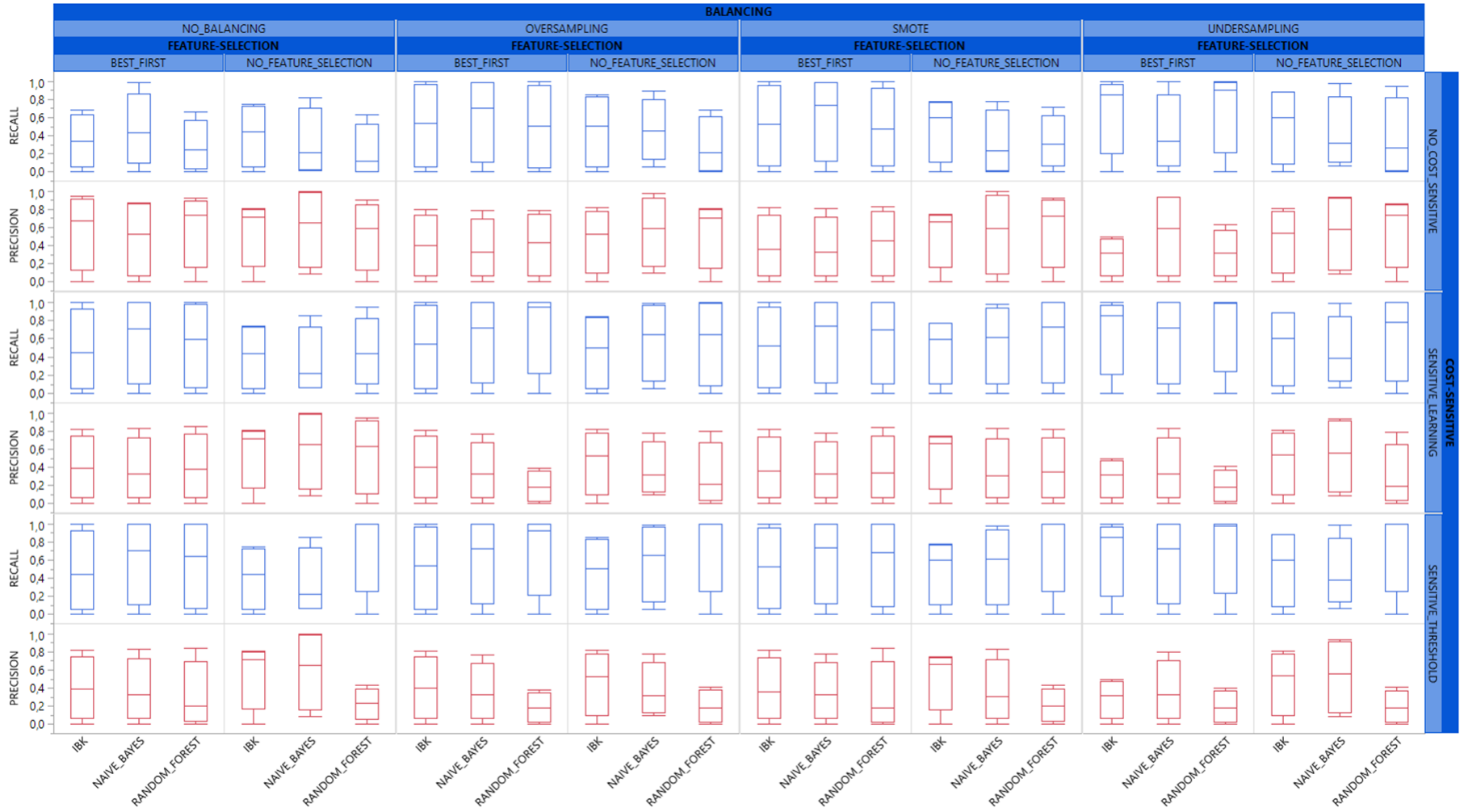
SAMPLING
- Oversampling
- Undersampling
- SMOTE

METRICHE ANALIZZATE
- Recall
- Kappa
- AUC
- Precision

La costruzione del modello predittivo e stata effettuata, combinando i classificatori con le varie tecniche tramite le **WEKA API** Java. Il dataset in formato .csv di output, invece è stato analizzato tramite il software statistico **JMP** che ha permesso una rappresentazione delle misure ottenute in formato grafico, tramite Box Plot.

RECALL e PRECISION - BOOKKEEPER

SOLO METRICHE DI CLASSE FORNITE

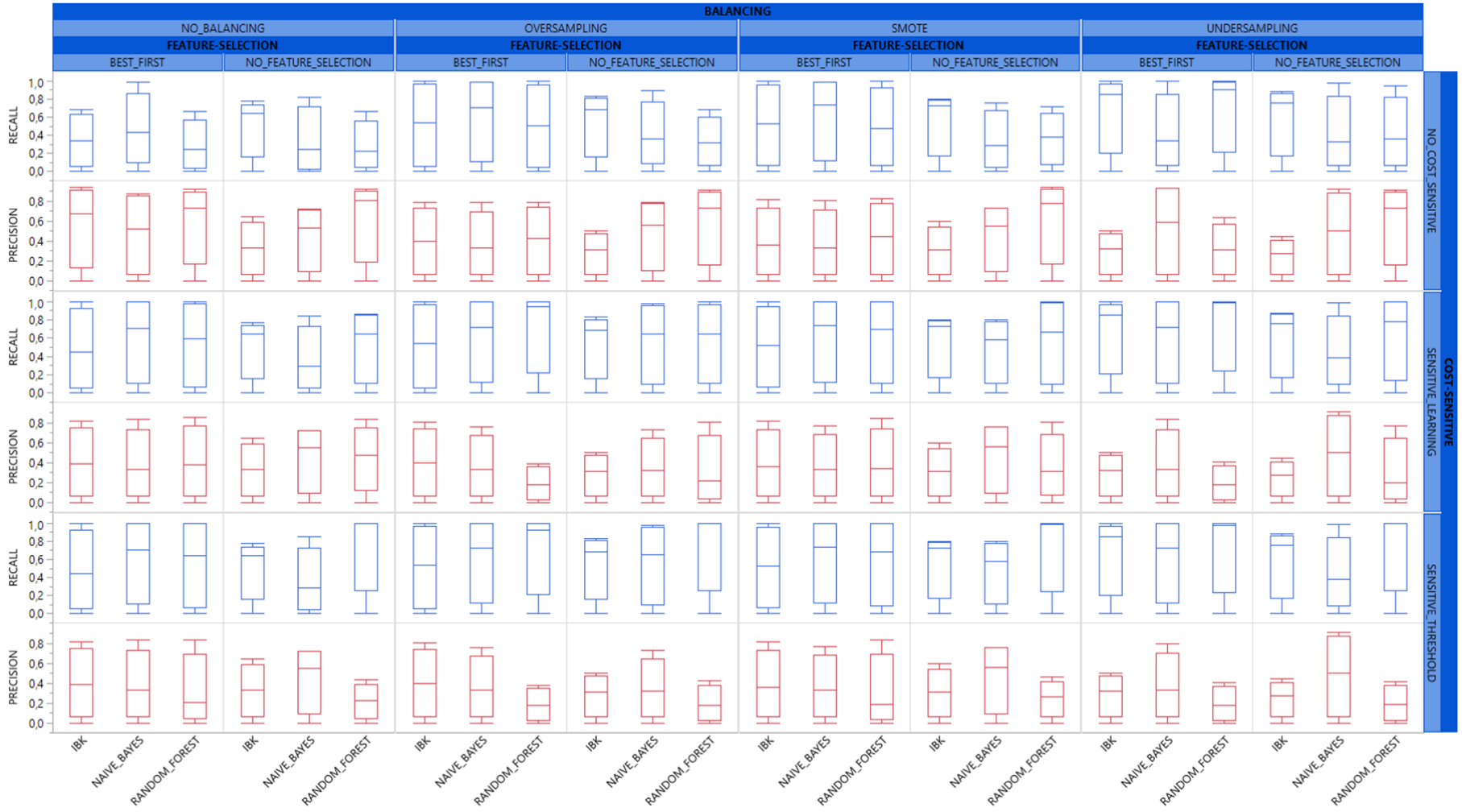


Si nota come senza l'utilizzo di tecniche di Cost Sensitive si ha una recall più bassa in particolar modo utilizzando il classificatore Random Forest, al contrario utilizzando tecniche di Cost Sensitive, si ha la recall migliore.

Inoltre, si nota come senza l'uso di Cost Sensitive si ha una precision più alta, in particolar modo con il classificatore Random Forest, e che con Cost Sensitive sia tra le più basse.

RECALL e PRECISION - BOOKKEEPER

CON METRICHE DI CLASSE AGGIUNTE: NLOCM + NPM

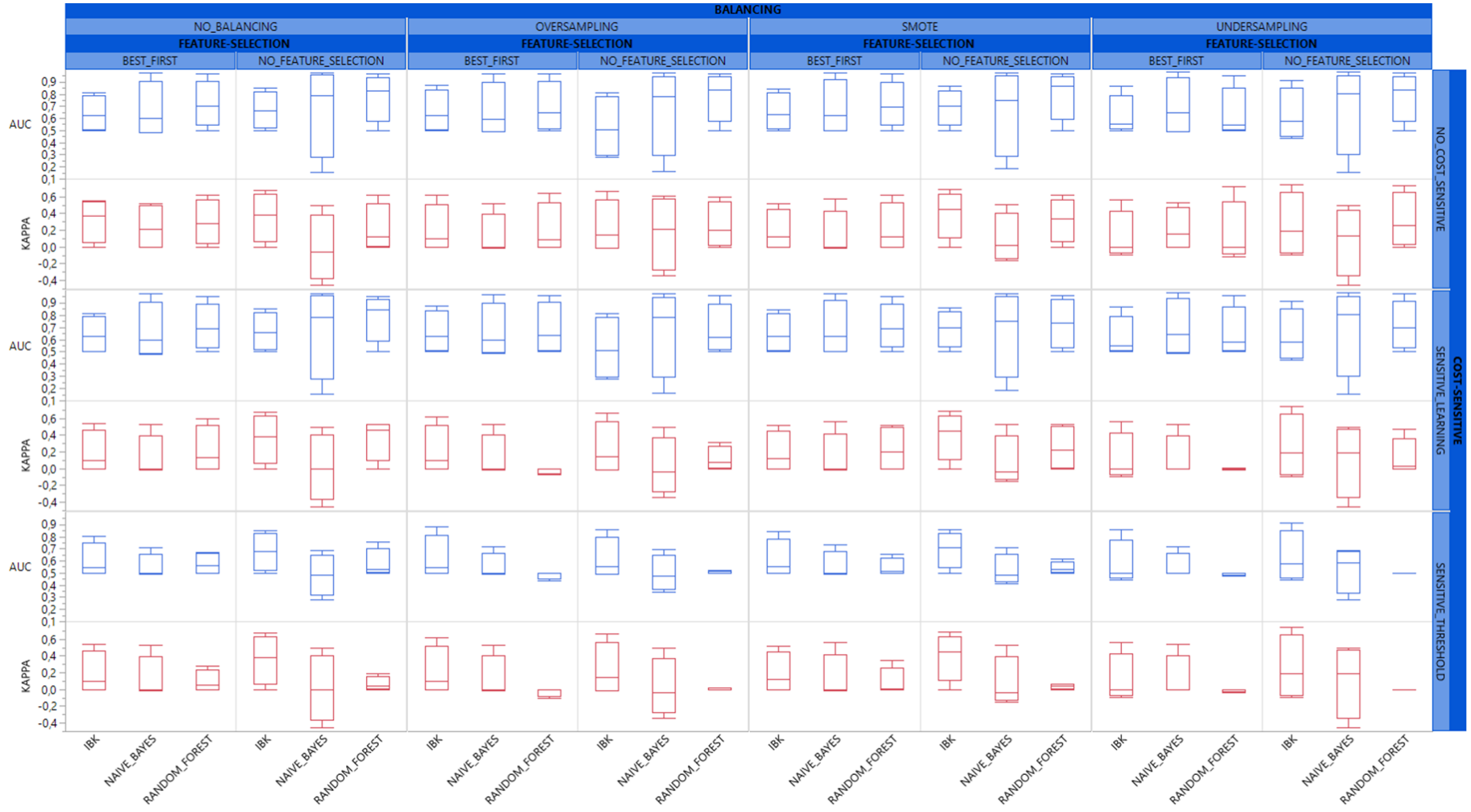


Prendendo in considerazione anche le metriche riguardanti il numero dei metodi pubblici e del numero di righe commentate all'interno della classe, si nota un miglioramento generale della recall, tranne che nelle combinazioni con Best First dove si hanno i medesimi valori della slide precedente.

Per la precision invece, troviamo una diminuzione generale, tranne nelle configurazioni con Undersampling o nelle configurazioni con Best First dove rimane invariata.

KAPPA e AUC - BOOKKEEPER

SOLO METRICHE DI CLASSE FORNITE

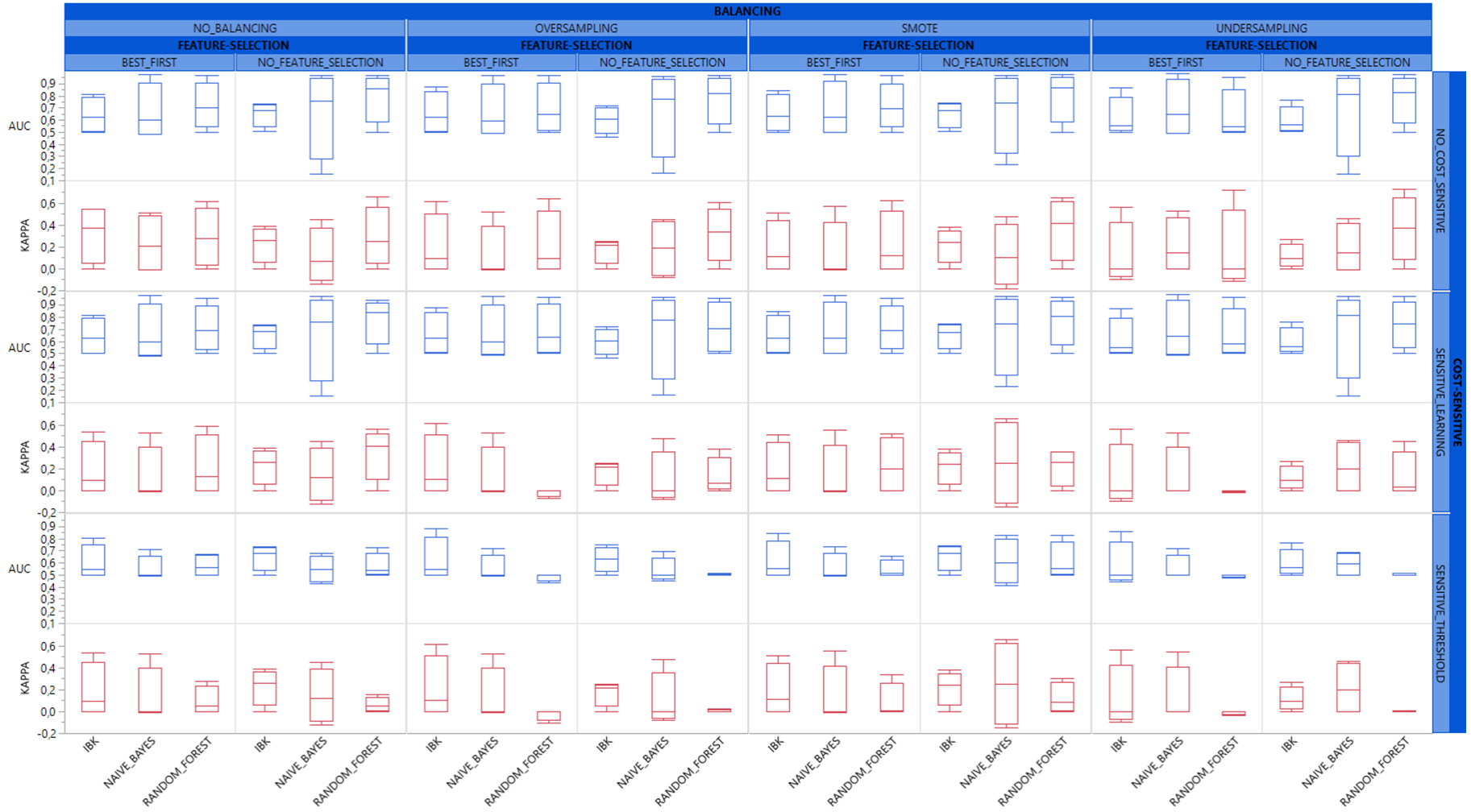


Si nota come si ha una AUC crescente utilizzando il classificatore Naive Bayes.

Inoltre si può notare come in generale il modello si comporti meglio rispetto uno dummy, in particolare tramite il classificatore IBk.

KAPPA e AUC - BOOKKEEPER

CON METRICHE DI CLASSE AGGIUNTE: NLOCM + NPM



Prendendo in considerazione anche le metriche riguardanti il numero dei metodi pubblici e del numero di righe commentate all'interno della classe, si nota delle misure molto simili per AUC.

Inoltre, si può notare come il modello, a differenza della slide precedente, ha generalmente una Kappa maggiore, in particolare si può notare come non scenda mai sotto il valore -0.2 con Cost Sensitive Threshold ed addirittura mai sotto lo 0 nelle altre configurazioni.

ANALISI RISULTATI

Per prima cosa si è proceduto ad analizzare le diverse combinazioni di classificatori e tecniche, con l'aggiunta o meno delle nuove metriche sviluppate. Si nota come, rispetto al caso di considerare solo le metriche fornite nel corso, aggiungendo singolarmente ogni metrica si è avuto un risultato diverso, infatti aggiungendo **NVPM** o **NSM** o **NAM** si è avuto una diminuzione generale della recall; invece, aggiungendo **NLOCM** o **NPM** si è avuto un aumento generale della recall.

Si è provato, inoltre, ad unire tutte le metriche in un singolo modello, ma con risultati mediocri molto sotto la media, ed unire le due metriche che hanno portato i risultati migliori, cioè numero dei metodi pubblici (**NPM**) + numero delle righe commentate (**NLOCM**), raggiungendo così la configurazione di metriche che ha portato l'aumento migliore della recall, a parità di mediana, in particolar modo per le configurazione Naive Bayes ed IBk, escludendo l'uso della tecnica Best First.

Comparando la configurazione, infatti, in cui si sono considerante anche tutte le metriche aggiunte con quella in cui vengono considerate solo le metriche fornite, si nota come utilizzando Best First si hanno le medesime misure, questo ci indica come le nuove metriche vengano tutte scartate dalla tecnica di Feature Selection. Inoltre, si è provato ad utilizzare Best First oltre che nella sua configurazione di default che fa utilizzo di Forward Search (parametro: "`-D 0`"), nella sua configurazione con Backward Search (parametro: "`-D 1`") producendo però i medesimi risultati, confermando valori di recall massima in configurazione senza Feature Selection.

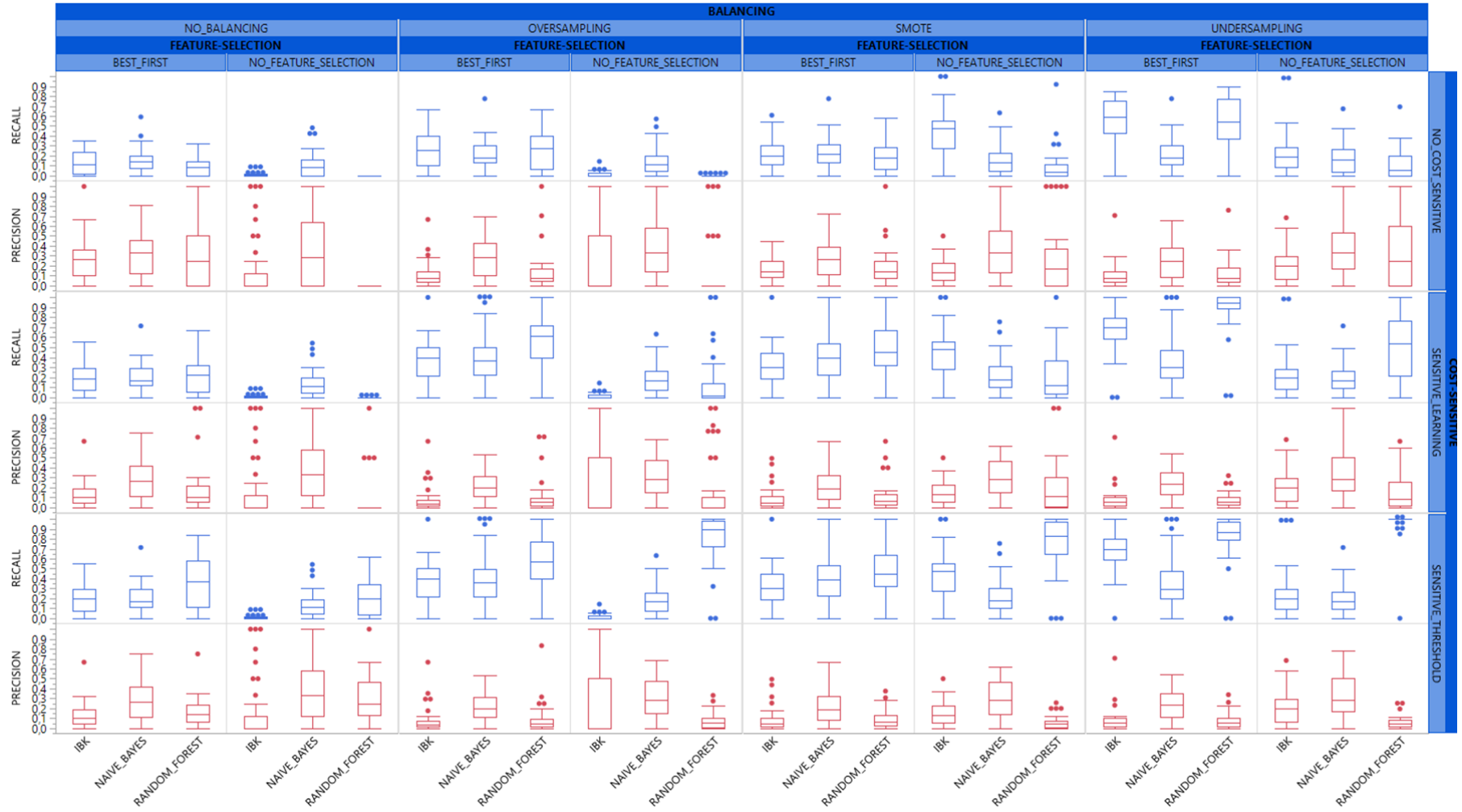
Essendo BookKeeper un progetto con poche release e di conseguenza generando un dataset di size minima, l'inutilizzo di Feature Selection e quindi l'utilizzo di un dataset con più feature non ha delle conseguenze impattanti sulla prestazioni della predizione e della fase di misurazione.

Analizzando il modello creato dall'unione delle metriche di classe fornite più quelle aggiunte **NPM + NLOCM**, si individua la miglior configurazione tra classificatori e tecniche utilizzate, a parità di mediana, e cioè:

RANDOM FOREST + COST SENSITIVE THRESHOLD

RECALL e PRECISION - SYNCOPE

SOLO METRICHE DI CLASSE FORNITE

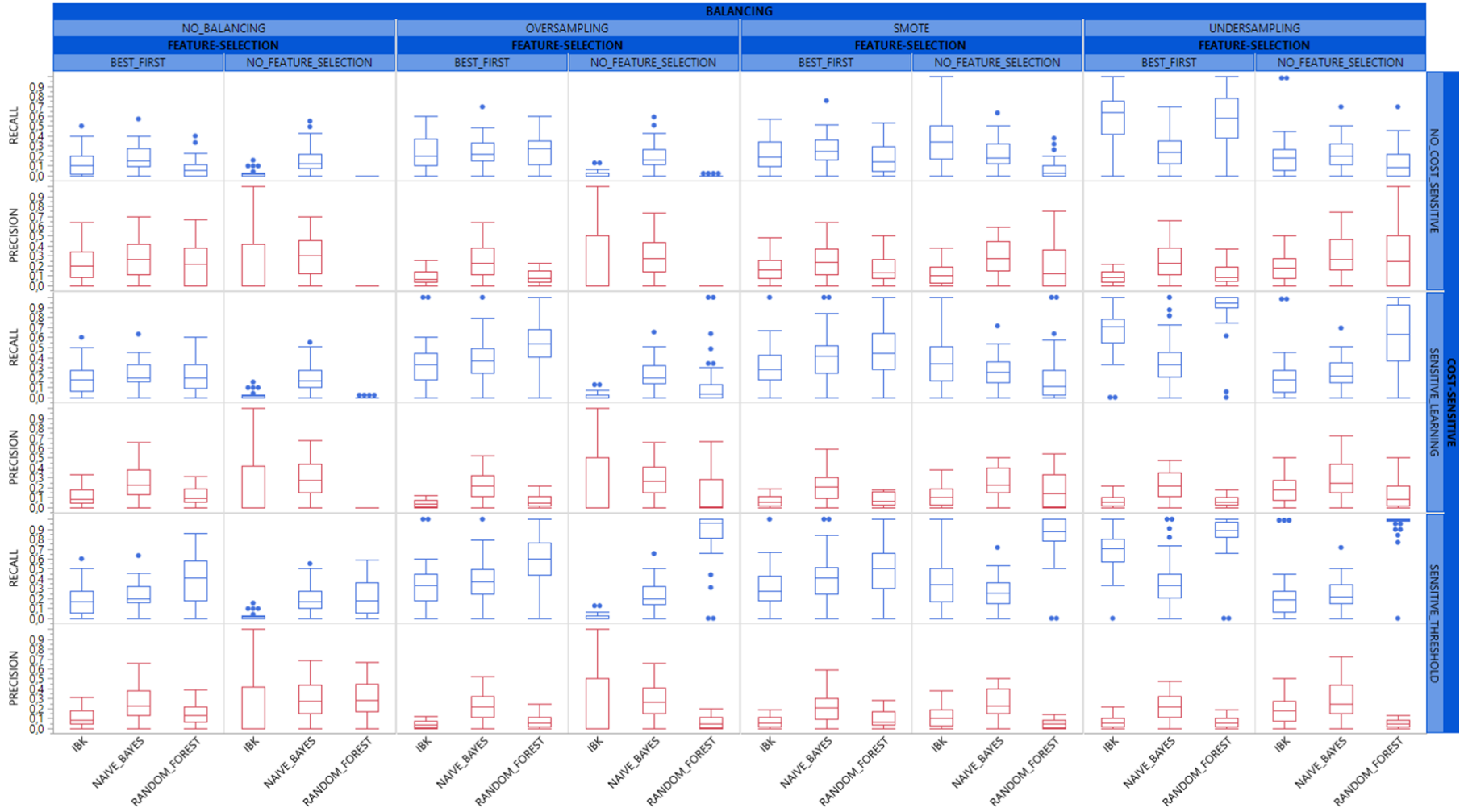


Si nota come l'utilizzo del classificatore Random Forest porti un aumento della recall.

Inoltre, si nota un aumento della precision utilizzando il classificatore Naive Bayes.

RECALL e PRECISION - SYNCOPE

CON METRICHE DI CLASSE AGGIUNTE: NPVM

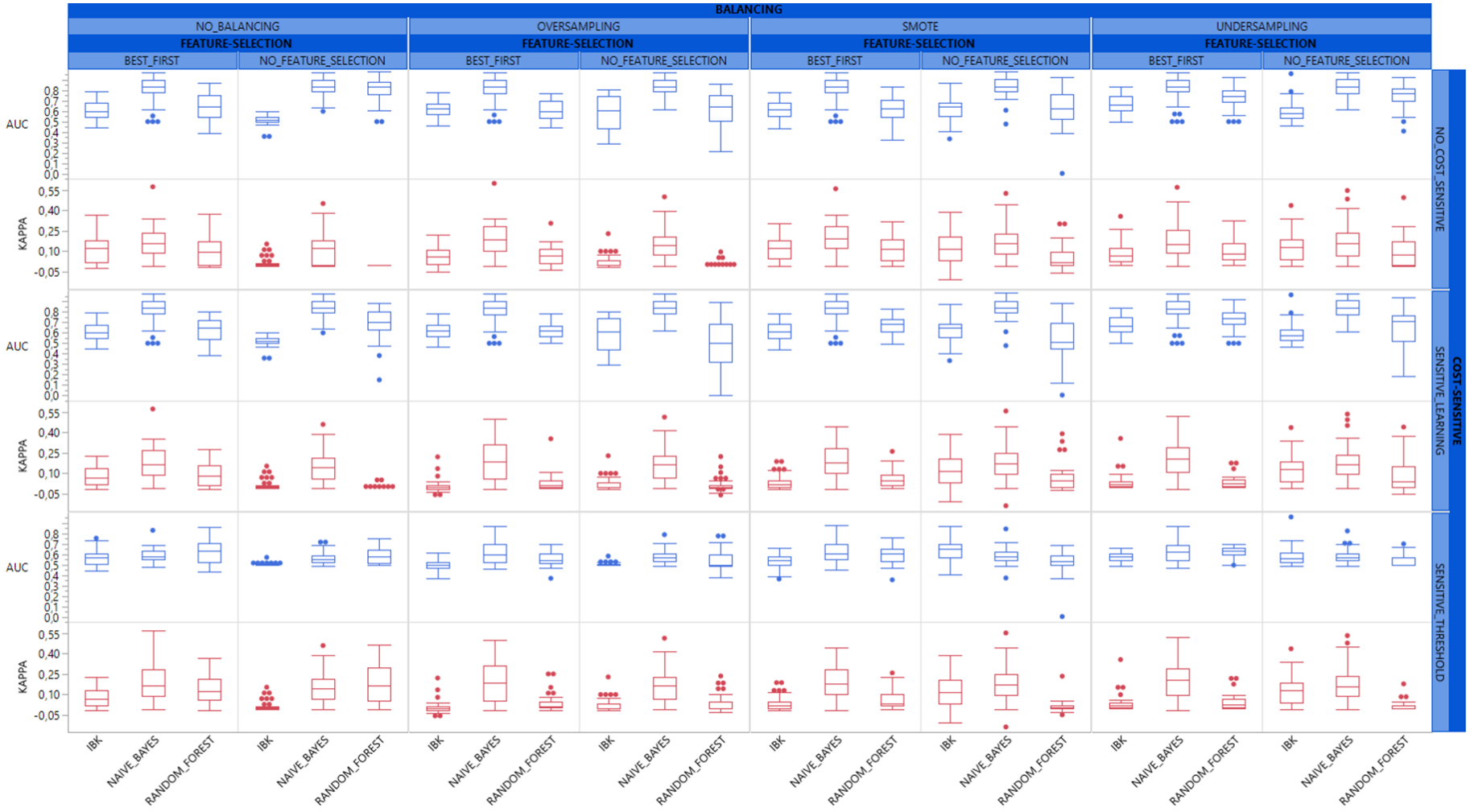


Prendendo in considerazione anche la metrica riguardante il numero di metodi privati all'interno della classe, si ha un miglioramento della recall, in particolare con il classificatore Random Forest.

Inoltre, si può notare una conseguente diminuzione della precision.

KAPPA e AUC - SYNCOPE

SOLO METRICHE DI CLASSE FORNITE

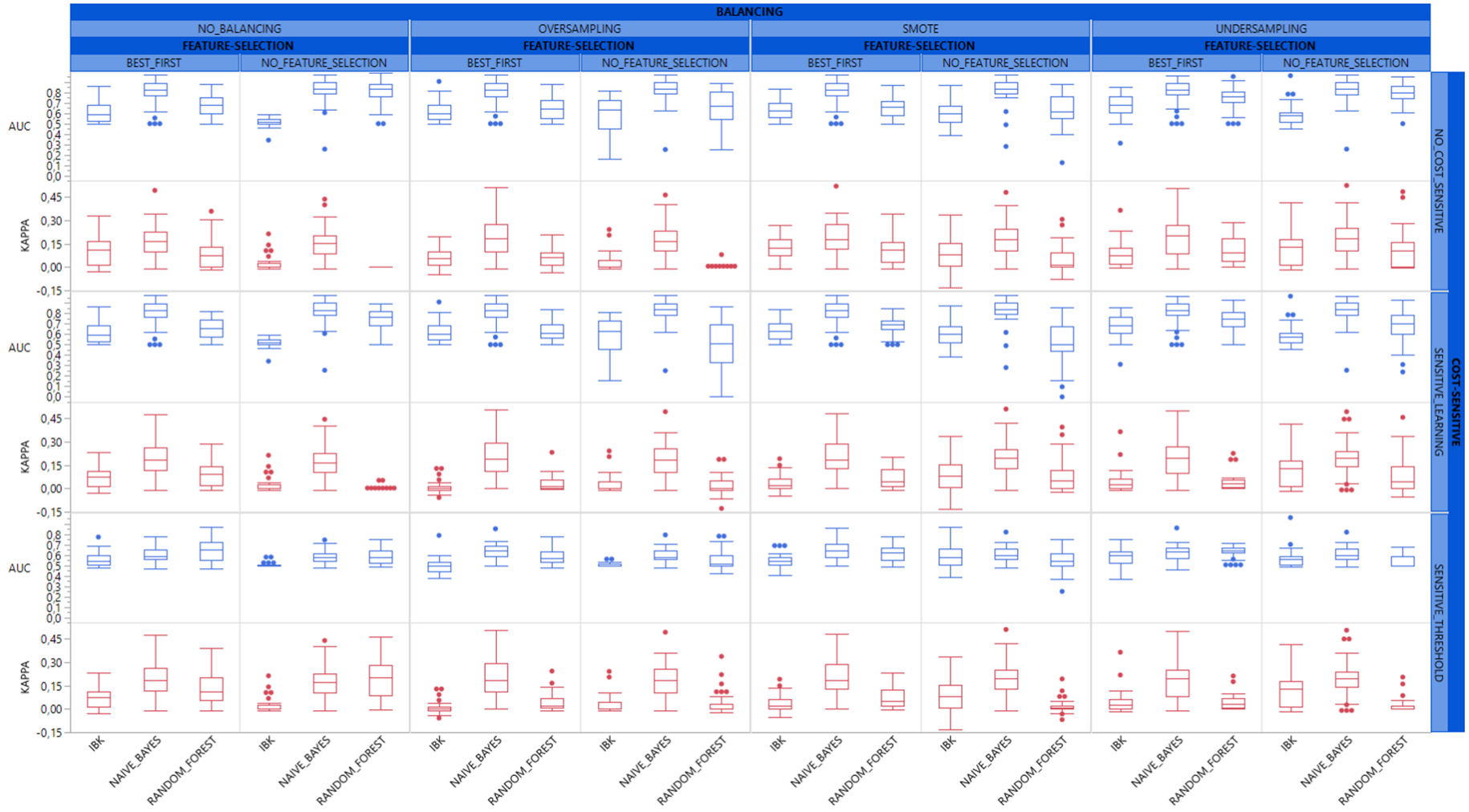


Si nota come, con il classificatore Naive Bayes si ha un aumento della AUC.

Inoltre, si può notare come le misure di Kappa non si discostano troppo dallo zero, infatti si può considerare il modello mediamente uguale o migliore di un modello dummy; all'interno del modello non compaiono mediane di Kappa al di sotto dello zero e si trova un aumento maggiore tramite il classificatore Naive Bayes.

KAPPA e AUC - SYNCOPE

CON METRICHE DI CLASSE AGGIUNTE: NPVM



Prendendo in considerazione anche la metrica riguardante il numero di metodi privati all'interno della classe, si ha una leggera diminuzione di AUC.

Inoltre, si può notare come il modello, a differenza della slide precedente, ha generalmente una Kappa maggiore tranne nei casi facenti uso delle tecniche SMOTE senza Feature Selection, dove addirittura peggiora.

ANALISI RISULTATI

Come per il progetto precedente, si è proceduto ad analizzare le diverse combinazioni di classificatori e tecniche, con l'aggiunta o meno delle nuove metriche sviluppate, in particolare analizzando l'aggiunta singola di ogni nuova metrica a quelle già fornite; infatti, aggiungendo **NSM** o **NLOCM** si è avuto un leggero peggioramento della recall, invece aggiungendo **NPM** o **NPVM** o **NAM** si è avuto un leggero miglioramento. Si è provato, inoltre, ad unire tutte le metriche in un singolo modello, ma con risultati sotto la media ed oltretutto, ad usare la configurazione ottima rilevata nel progetto precedente, che in questo caso però non ha portato miglioramenti.

A differenza di BookKeeper, analizzando le misure del progetto Syncope si nota un'alta presenza di valori outliers, in particolar modo nella configurazione di metriche che non prende in considerazione quelle aggiuntive. Tra le configurazioni di metriche si è scelto come la migliore quella con aggiunta **NPVM**, che oltre ad un aumento della recall ha portato un alta omogeneità nelle misure; omogeneità raggiunta in generale anche grazie l'utilizzo della tecnica di Feature Selection Best First. Si nota inoltre che tra le metriche aggiuntive, solo due non vengono scartate da Best First e cioè **NPVM** e **NLOCM**.

Analizzando il modello creato dall'unione delle metriche di classe fornite più quella aggiunta **NPVM** si individuano 2 configurazioni ottime, ognuna con i propri **PRO** e **CONTRO**:

- **RANDOM FOREST + BEST FIRST + UNDERSAMPLING + COST SENSITIVE LEARNING**: alta recall ma essendo il dataset complesso, potrebbe portare, utilizzando Undersampling, ad un modello con dati non realmente rappresentativi;
- **RANDOM FOREST + OVERSAMPLING + COST SENSITIVE THRESHOLD**: stesso livello alto di recall, ma senza l'utilizzo di Feature Selection, scelta che potrebbe impattare sulle prestazioni di misurazione.

In conclusione, se di poco conto le prestazioni di misurazione, si predilige la seconda configurazione così da avere delle predizioni più accurate.

LINK UTILI

- PROGETTO GITHUB:
https://github.com/callbrok/BugRetriever

- SONARCLOUD:
https://sonarcloud.io/summary/overall?id=callbrok_BugRetriever

- RIFERIMENTI BIBLIOGRAFICI:
https://www.synopsys.com/content/dam/synopsys/sig-assets/reports/cpsq-report-nov-22-1.pdf

GRAZIE PER L'ATTENZIONE

FINE PRESENTAZIONE