

CANCRO AL SENO: ONTOLOGIA, PREDIZIONE E PRENOTAZIONE

Giacovazzo Christian – 716696 – c.giacovazzo1@studenti.uniba.it

Progetto per “Ingegneria della conoscenza” 2021/2022

GitHub: https://github.com/Christian-is/ICon2122_progetto_breast_cancer

INDICE

1. INTRODUZIONE	3
2. ONTOLOGIA	4
3. APPRENDIMENTO SUPERVISIONATO.....	6
4. APPRENDIMENTO NON SUPERVISIONATO	12
5. APPRENDIMENTO: CONCLUSIONI	13
6. CSP	14

1. INTRODUZIONE

Il progetto nasce con l'intento di predire se un paziente possa ripresentare in futuro un cancro al seno. A tal proposito sono state adottate una serie di tecniche sia di apprendimento supervisionato sia non supervisionato. Inoltre, è stata modellata un'ontologia di dominio che offre una rappresentazione formale e concettualizzata della realtà presa in esame, affinché possa venire interrogata. L'ontologia è completa di individui d'esempio in modo tale da testarne la funzionalità. Infine è stato sviluppato un sistema di prenotazioni per una ipotetica visita istologica presso un laboratorio di analisi.

Il dataset utilizzato contiene le seguenti features:

1. **Class [target]:** no-recurrence-events, recurrence-events
2. **age:** 10-19, 20-29, 30-39, 40-49, 50-59, 60-69, 70-79, 80-89, 90-99
3. **menopause:** lt40, ge40, premeno
4. **tumor-size:** 0-4, 5-9, 10-14, 15-19, 20-24, 25-29, 30-34, 35-39, 40-44, 45-49, 50-54, 55-59
5. **inv-nodes:** 0-2, 3-5, 6-8, 9-11, 12-14, 15-17, 18-20, 21-23, 24-26, 27-29, 30-32, 33-35, 36-39
6. **node-caps:** yes, no
7. **deg-malig:** 1, 2, 3
8. **breast:** left, right
9. **breast-quad:** left-up, left-low, right-up, right-low, central
10. **irradiat:** yes, no

Per la modellazione dell'ontologia è stato utilizzato Protégé, mentre il codice è stato scritto interamente in Python. È possibile visualizzare il contenuto dell'ontologia tramite OwlReady2 direttamente su Python. I classificatori sono stati sviluppati con le librerie Scikit-learn, Pandas e Seaborn. Il CSP per il sistema di prenotazione è stato sviluppato con l'utilizzo di Constraint.

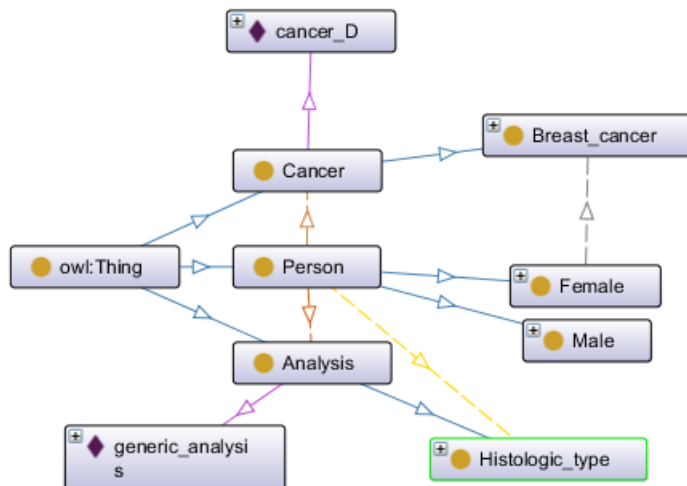
Fonti del materiale utilizzato:

- Python: <https://www.anaconda.com/products/individual>
- Dataset: <https://archive-beta.ics.uci.edu/ml/datasets/breast+cancer>
- Protégé: <https://protege.stanford.edu/>
- Scikit-learn: <https://scikit-learn.org/stable/install.html>
- Pandas: <https://pandas.pydata.org/>
- Seaborn: <https://seaborn.pydata.org/>
- OwlReady2: <https://owlready2.readthedocs.io/en/v0.37/index.html>
- Constraint: <https://pypi.org/project/python-constraint/>

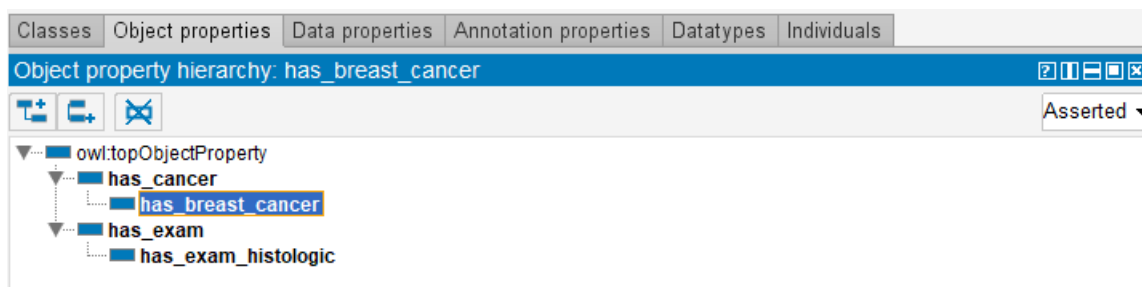
2. ONTOLOGIA

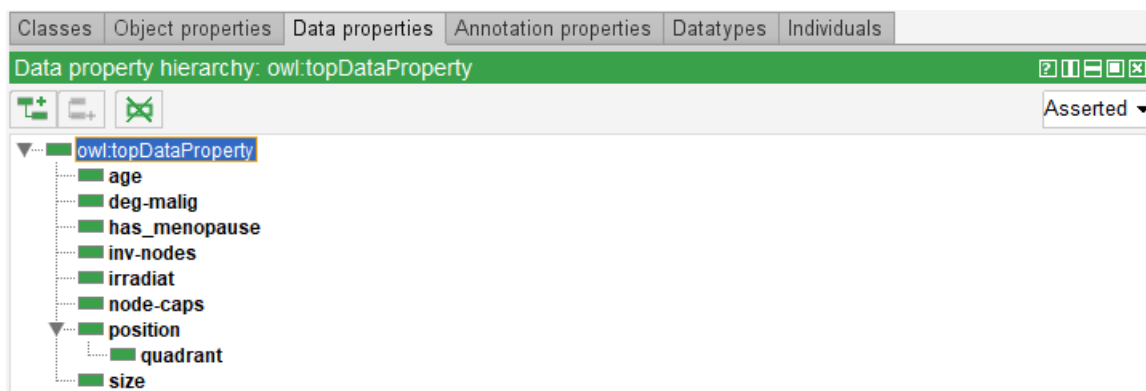
Un' ontologia è la specificazione dei significati dei simboli in un sistema informatico. La specifica formale è importante per l'interoperabilità semantica, ovvero l'abilità di basi di conoscenza differenti di operare insieme ad un livello semantico tale che i significati dei simboli sono rispettati. L'ontologia viene descritta come "una specificazione di una concettualizzazione". Una rappresentazione formale di un insieme di conoscenze è una concettualizzazione, ossia un insieme di oggetti, concetti e relazioni fra di essi che esistono in una particolare area d'interesse. Nella modellazione dell' ontologia è stato usato il programma Protégé. Sono state aggiunte delle proprietà di oggetto e di dati con un dominio di appartenenza e un loro range, mantenendo una certa coerenza con il dataset di riferimento utilizzato. È possibile interrogare l'ontologia e ottenere informazioni usando il tab di Protégé "DL query". Si è usato come reasoner HermiT, di default in Protégé.

Di seguito la modellazione della ontologia.

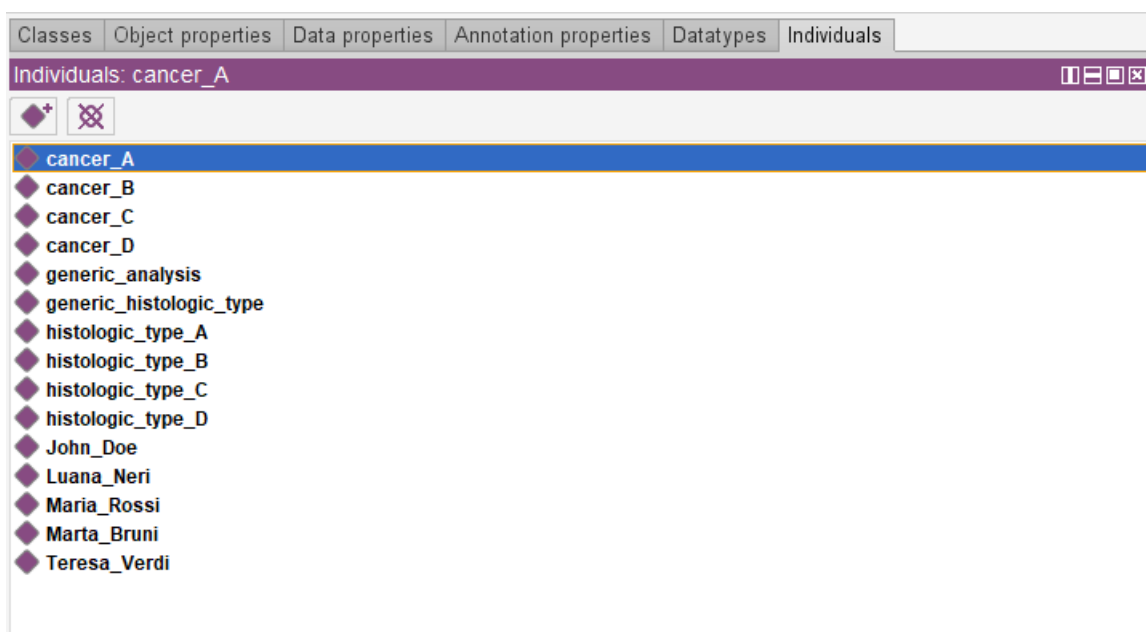


Queste entità sono in relazione tra di loro grazie a delle proprietà sia di oggetto che di data. Una object property permette di mettere in relazione due individui, siano essi di classi distinte o della stessa classe. Una data property permette di mettere in relazione un individuo con un valore di tipo primitivo.





Inoltre, per alcune entità si sono create delle istanze. Alcune ad esempio per individuare una tipologia di esame come quella istologica, altre per individuare istanze di persone a cui attribuire un cancro e le relazioni con esso.



Successivamente sono state formulate delle query per interrogare l'ontologia. Si è partiti con query semplici come cercare persone affette da un tumore qualsiasi o specifico al seno e a quali esami istologici si sono sottoposte, fino a query più complesse con la combinazione di più attributi inerenti a ciascuna persona.

DL query:	DL query:	DL query:
Query (class expression) Person that has_cancer some [Execute] [Add to ontology]	Query (class expression) Person that has_breast_cancer some [Execute] [Add to ontology]	Query (class expression) Person that has_exam_histologic value histologic_type_B [Execute] [Add to ontology]
Query results Instances (5 of 5) ◆ John_Doe ◆ Luana_Neri ◆ Maria_Rossi ◆ Marta_Bruni ◆ Teresa_Verdi	Query results Instances (4 of 4) ◆ Luana_Neri ◆ Maria_Rossi ◆ Marta_Bruni ◆ Teresa_Verdi	Query results Instances (2 of 2) ◆ Maria_Rossi ◆ Teresa_Verdi

DL query:

Query (class expression)


Person **that** has_exam_histologic **value** histologic_type_B **and** has_menopause **value** "ig40"

Execute

Add to ontology

Query results

Instances (1 of 1)

 Maria_Rossi

L'ontologia è consultabile anche attraverso la libreria Python OwlReady2.

```

1 # -*- coding: utf-8 -*-
2
3 import owlready2 as owl
4 import os
5
6 def ontology_analyzer():
7     print("ONTOLOGIA\n")
8     onto = owl.get_ontology("breast_ontology.owl").load()
9
10    #stampo il contenuto principale della ontologia
11    print("Lista classi nella ontologia:\n")
12    print(list(onto.classes()), "\n")
13
14    print("Lista Person nella ontologia:\n")
15    persons = onto.search(is_a = onto.Person)
16    print(persons, "\n")
17
18    print("Lista Cancer nella ontologia:\n")
19    cancers = onto.search(is_a = onto.Cancer)
20    print(cancers, "\n")
21
22    print("Lista Breast_cancer nella ontologia:\n")
23    b_cancers = onto.search(is_a = onto.Breast_cancer)
24    print(b_cancers, "\n")
25
26    print("Lista Analysis nella ontologia:\n")
27    analysis = onto.search(is_a = onto.Analysis)
28    print(analysis, "\n")
29
30    #esempio di query
31    print("Lista di persone che hanno un cancro al seno:\n")
32    patients = onto.search(is_a = onto.Person, has_breast_cancer = "")
33    print(patients, "\n")
34
  
```

Lista classi nella ontologia:

```
[breast_ontology.Analysis, breast_ontology.Histologic_type,
breast_ontology.Breast_cancer, breast_ontology.Cancer, breast_ontology.Female,
breast_ontology.Male, breast_ontology.Person]
```

Lista Person nella ontologia:

```
[breast_ontology.Person, breast_ontology.Female, breast_ontology.Male,
breast_ontology.Luana_Neri, breast_ontology.Marta_Bruni,
breast_ontology.Teresa_Verdi, breast_ontology.Maria_Rossi,
breast_ontology.John_Doe]
```

Lista Cancer nella ontologia:

```
[breast_ontology.Cancer, breast_ontology.cancer_D, breast_ontology.Breast_cancer,
breast_ontology.cancer_B, breast_ontology.cancer_C, breast_ontology.cancer_A]
```

Lista Breast_cancer nella ontologia:

```
[breast_ontology.Breast_cancer, breast_ontology.cancer_B,
breast_ontology.cancer_C, breast_ontology.cancer_A]
```

Lista Analysis nella ontologia:

```
[breast_ontology.Analysis, breast_ontology.generic_analysis,
breast_ontology.Histologic_type, breast_ontology.generic_histologic_type,
breast_ontology.histologic_type_A, breast_ontology.histologic_type_B,
breast_ontology.histologic_type_C, breast_ontology.histologic_type_D]
```

Lista di persone che hanno un cancro al seno:

```
[breast_ontology.Luana_Neri, breast_ontology.Marta_Bruni,
breast_ontology.Teresa_Verdi, breast_ontology.Maria_Rossi]
```

3. APPRENDIMENTO SUPERVISIONATO

Con l'Apprendimento Supervisionato si cerca di costruire un modello partendo da dei dati di addestramento etichettati, con i quali si fanno previsioni su dati non disponibili o futuri.

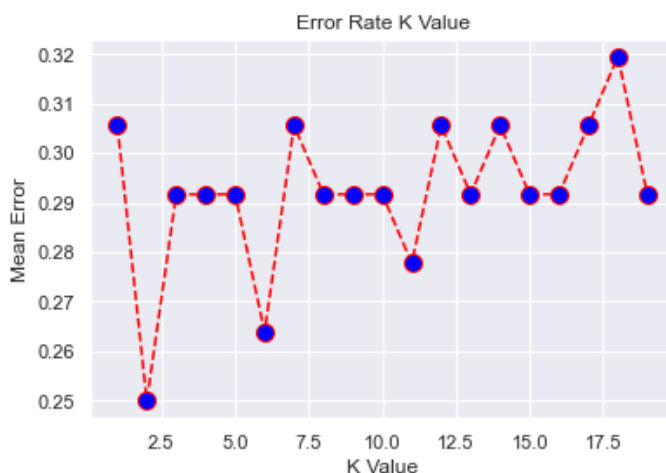
Per ogni algoritmo di apprendimento supervisionato sono stati prodotti i seguenti grafici:

- ROC Curve
- Precision-Recall Curve
- Matrice di Confusione

Per la maggior parte degli algoritmi è stata usata la tecnica della cross-validation per rilevare possibili problemi di overfitting. A tal proposito si riportano i valori del punteggio medio (cross-val-score), della varianza, deviazione standard su cinque iterazioni.

3.1 K-nearest-neighbour

Il KNN è un algoritmo di apprendimento supervisionato che consiste nell'individuare i k esempi più vicini a quello che si intende classificare, a quest'ultimo viene quindi attribuita la categoria "più ricorrente" tra i k esempi più vicini.



Il grafico proposto fornisce un suggerimento sulla scelta del numero di vicini per minimizzare l'errore medio. Si evince che con numero di vicini uguale a 2 l'errore medio minimo commesso è di 0.25.

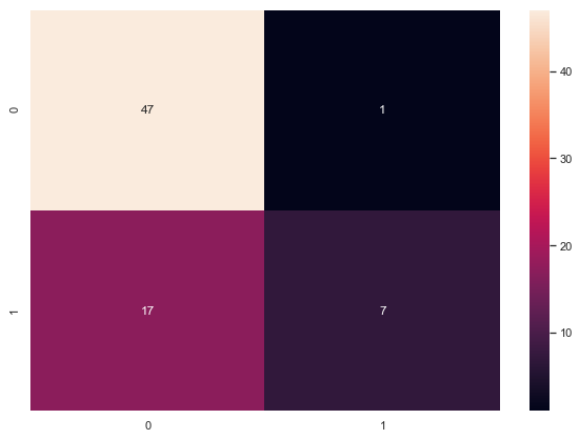
Dalla classificazione dell'algoritmo sono stati prodotti i seguenti classification report.

Clasification report:				
	precision	recall	f1-score	support
0	0.73	0.98	0.84	48
1	0.88	0.29	0.44	24
accuracy			0.75	72
macro avg	0.80	0.64	0.64	72
weighted avg	0.78	0.75	0.71	72

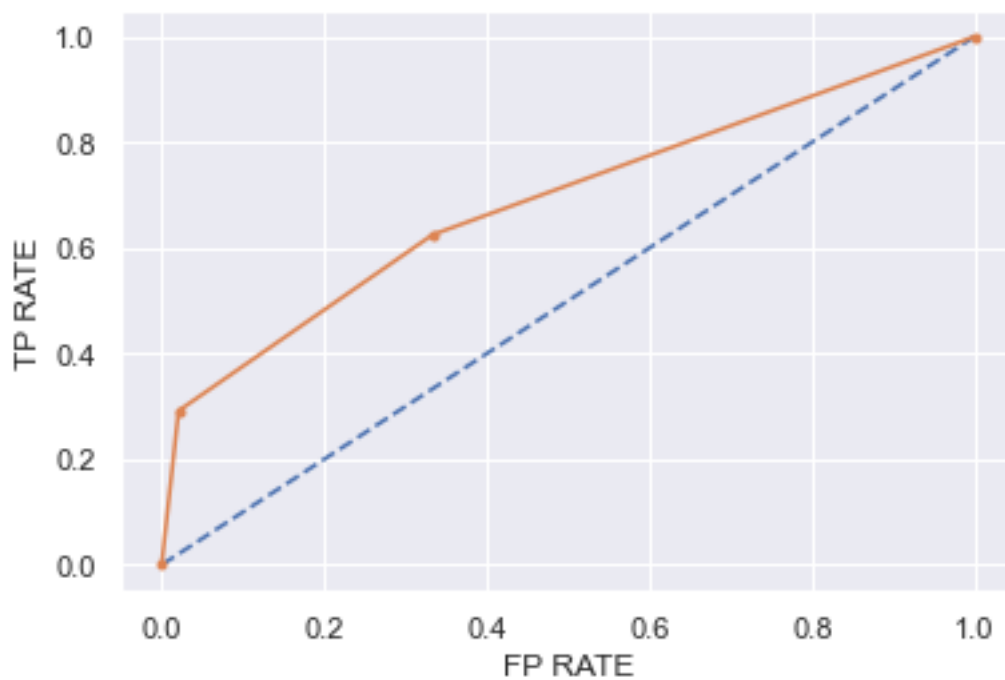
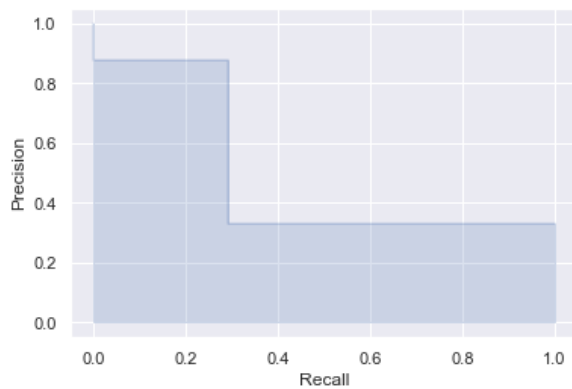
Si osservi che l'accuratezza dell'algoritmo è 0.75

Inoltre l'average precision è di 0.49

Confusion-Matrix:



Precision e Recall:



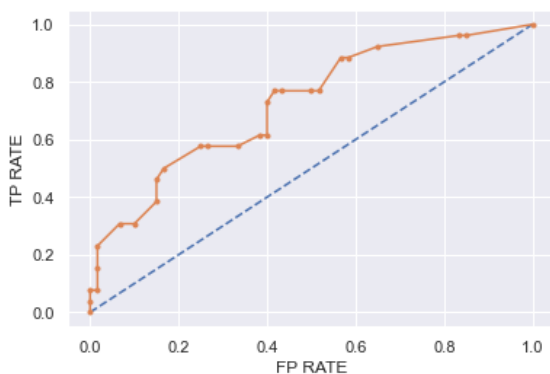
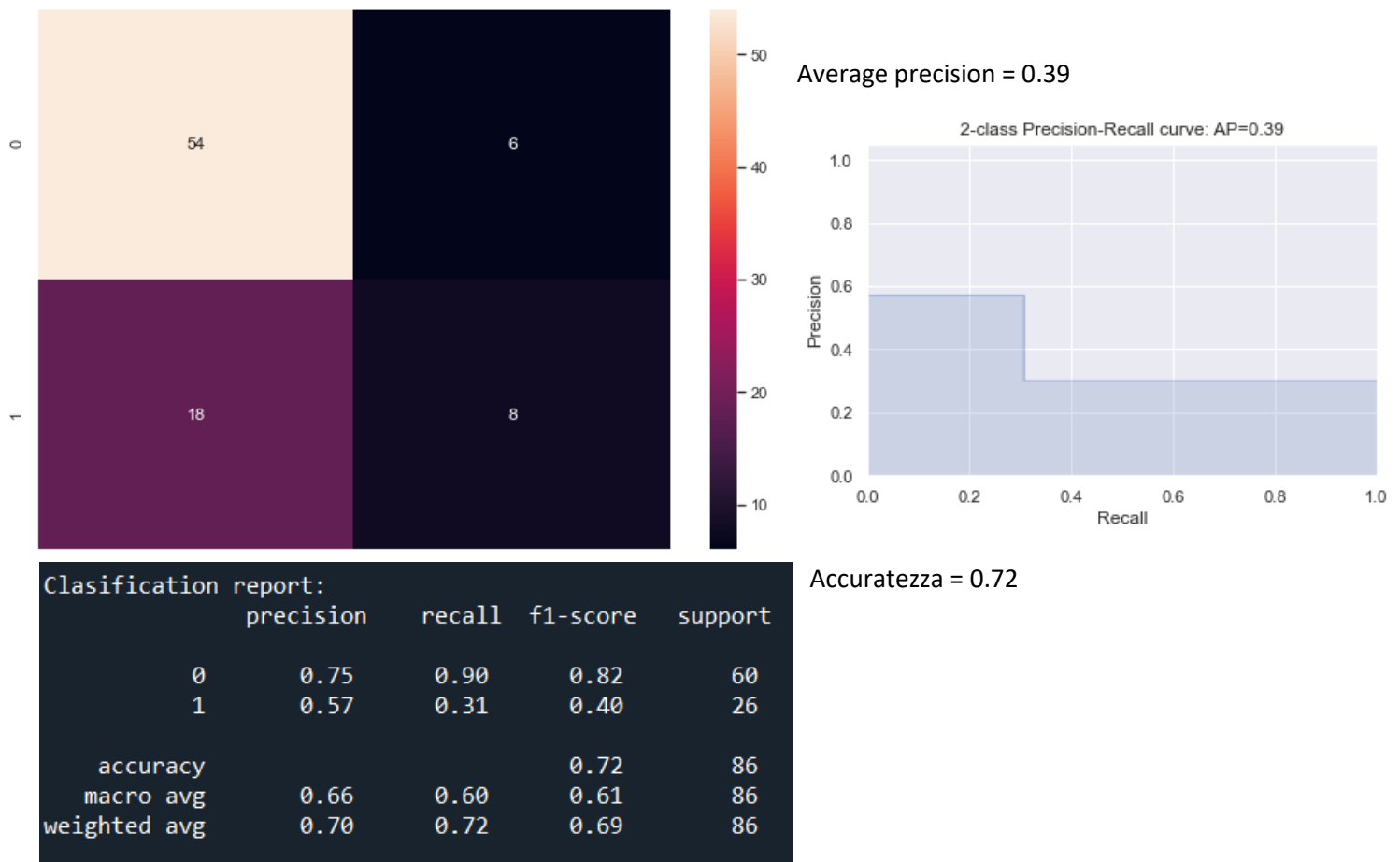
La curva ROC (Receiver Operating Characteristics) dell'AUC (Area Under The Curve) è una delle metriche di valutazione più importanti per il controllo delle prestazioni di qualsiasi modello di classificazione. Il ROC è una curva di probabilità e l'AUC rappresenta la misura della separabilità. Indica quanti modelli è in grado di distinguere tra le classi. Maggiore è l'AUC, migliore è il modello nel predire 0 come 0 e 1 come 1 (ovvero “no-recurrence-events” o “recurrence-events” nel caso in esame). Nel nostro caso l'AUC è pari a 0.688.

Per quanto riguarda la cross validation sul classificatore del KNN i dati ottenuti sono:

```
cv_scores mean:0.7169388989715668
cv_score variance:0.0011727526881956535
cv_score dev standard:0.03424547690127345
```


3.2 Random Forest

Il random forest è un modello composito costituito da molti alberi di decisione, ognuno dei quali fornisce una predizione. Esse vengono poi combinate allo scopo di ottenere una previsione complessiva della foresta per un dato esempio. La predizione di ciascun albero può essere ottenuta o attraverso la media delle predizioni di un albero per ogni esempio, oppure attraverso un meccanismo di votazione in cui tutti gli alberi votano la propria classificazione più probabile e l'esempio col maggior numero di voti sarà scelto come predizione finale.



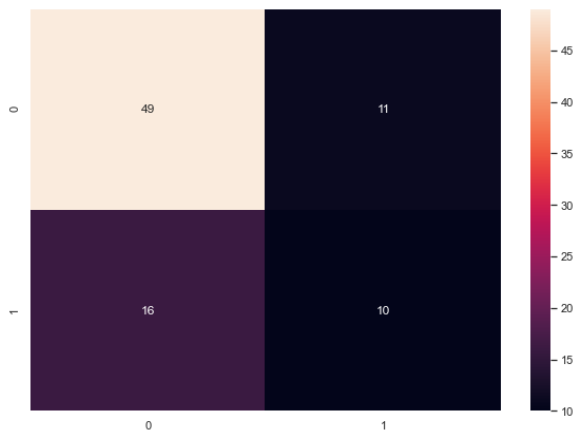
L'algoritmo Random Forest, secondo la curva ROC, è in grado di differenziare meglio le due classi rispetto al Knn; il valore AUC infatti è pari a 0.725.

Invece, per quanto riguarda la cross-validation i dati sono:

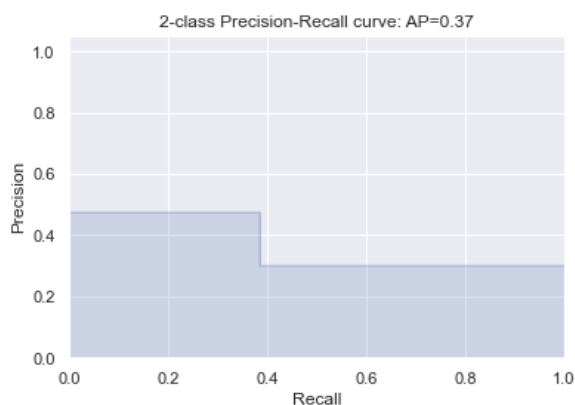
```
cv_scores mean:0.6993345432546885
cv_score variance:0.004824899932623555
cv_score dev standard:0.06946149964277733
```

3.3 Multinomial Naive Bayes

I classificatori “Naive Bayes” sono una famiglia di semplici classificatori probabilistici basati sull’applicazione del teorema di Bayes con una forte assunzione di indipendenza tra le feature. Con un modello di eventi multinomiali, gli esempi (vettori di feature) rappresentano le frequenze con cui certi eventi sono stati generati da una distribuzione polinomiale (p_1, \dots, p_n) dove p_i è la probabilità che l’evento i si verifichi.

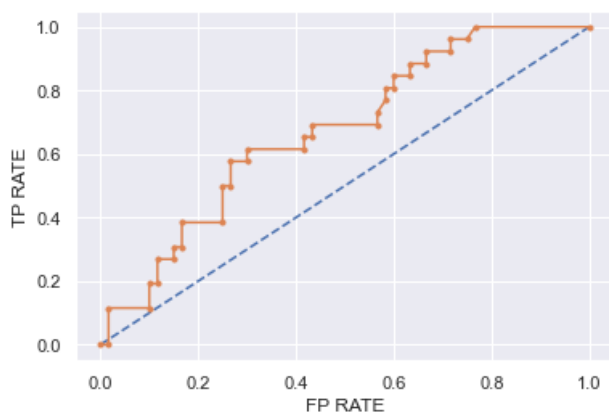


Clasification report:				
	precision	recall	f1-score	support
0	0.75	0.82	0.78	60
1	0.48	0.38	0.43	26
accuracy			0.69	86
macro avg	0.62	0.60	0.60	86
weighted avg	0.67	0.69	0.68	86



Il Multinomial Naive Bayes propone un’accuratezza di 0.69 e average precision di 0.37.

Come anche osservato dal grafico a sinistra, tra gli algoritmi trattati fin ora, questo è quello che propone una precision (al variare della recall) leggermente più bassa.



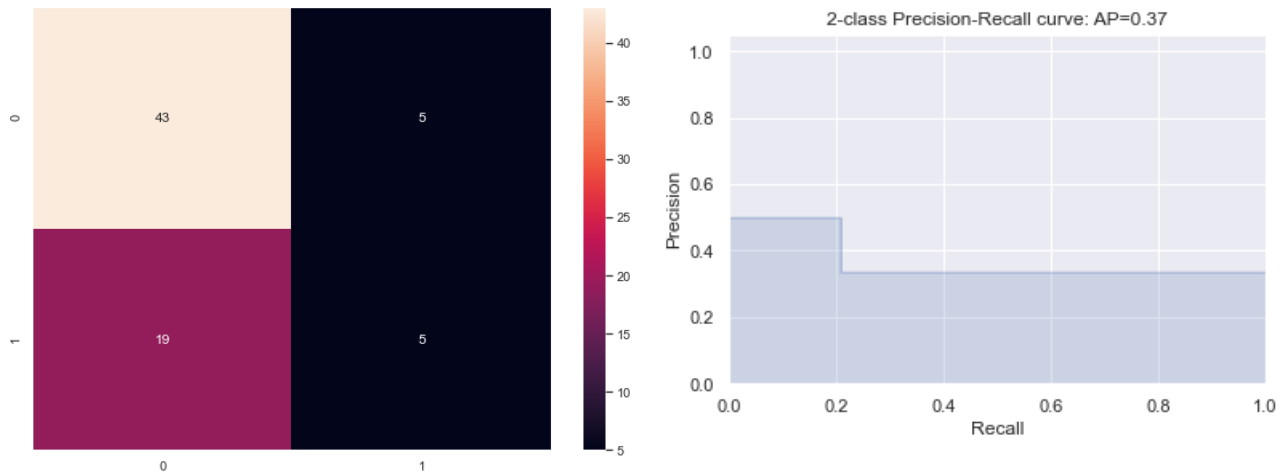
Per quanto riguarda l’AUC, esso è 0.673.

Per quanto riguarda la cross-validation i dati sono:

```
cv_scores mean:0.6782214156079854
cv_score variance:0.016379773306265645
cv_score dev standard:0.1279834884126294
```

3.4 Support-Vector Machines

Un modello SVM è una rappresentazione degli esempi come punti nello spazio, mappati in modo tale che gli esempi appartenenti alle due diverse categorie siano chiaramente separati da uno spazio il più possibile ampio. I nuovi esempi sono quindi mappati nello stesso spazio e la predizione della categoria alla quale appartengono viene fatta sulla base del lato nel quale ricade.



```
Clasification report:
      precision    recall  f1-score   support

     0       0.69      0.90      0.78        48
     1       0.50      0.21      0.29        24

 accuracy      0.67              72
 macro avg      0.60      0.55      0.54        72
 weighted avg      0.63      0.67      0.62        72

cv_scores mean:0.7027828191167573
cv_score variance:0.003328330421982946
cv_score dev standard:0.05769168416663658
```

L'algoritmo riporta una accuratezza di 0.67 e una average precision di 0.37

I dati sulla cross validation sono a sinistra.

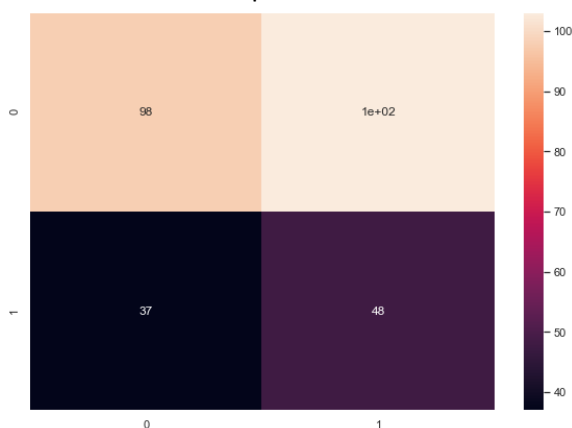
Questo algoritmo sembra avere il comportamento peggiore in questo dataset, in quanto ha accuratezza e average precision molto bassi rispetto agli altri esaminati.

4. APPRENDIMENTO NON SUPERVISIONATO

L'apprendimento non supervisionato è una tecnica di apprendimento automatico che consiste nel fornire al sistema una serie di input che esso riclassificherà ed organizzerà sulla base di caratteristiche comuni per cercare di effettuare ragionamenti e previsioni sugli input successivi. Al contrario dell'apprendimento supervisionato, durante l'apprendimento vengono forniti all'algoritmo solo esempi non etichettati, in quanto le classi non sono note a priori ma devono essere apprese automaticamente.

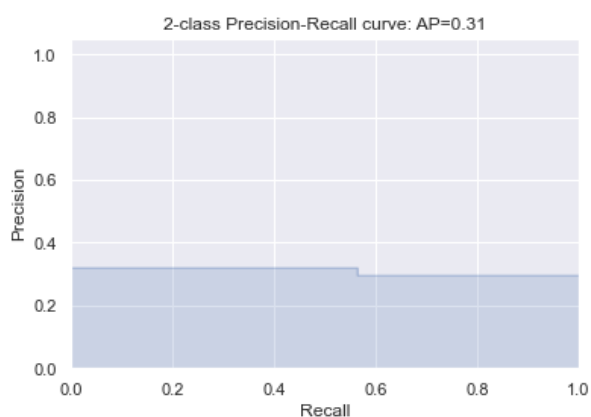
4.1 K-Means

L'algoritmo K-means è un algoritmo di hard-clustering partizionale che permette di suddividere un insieme di oggetti in K gruppi (nel nostro caso $K=2$) sulla base dei loro attributi. Si assume che gli attributi degli oggetti possano essere rappresentati come vettori, e che quindi formano uno spazio vettoriale. Ogni cluster viene identificato mediante un centroide. L'algoritmo segue una procedura iterativa. Inizialmente crea K partizioni e assegna ad ogni partizione i punti d'ingresso (casualmente o usando alcune informazioni euristiche). Successivamente calcola il centroide di ogni gruppo. Costruisce quindi una nuova partizione associando ogni punto d'ingresso al cluster il cui centroide è più vicino ad esso. Dopodiché vengono ricalcolati i centroidi per i nuovi cluster. Si itera il processo finché l'algoritmo non converge.



Clasification report:				
	precision	recall	f1-score	support
0	0.73	0.49	0.58	201
1	0.32	0.56	0.41	85
accuracy			0.51	286
macro avg	0.52	0.53	0.50	286
weighted avg	0.60	0.51	0.53	286

L'accuratezza è 0.51, la più bassa di tutti gli algoritmi usati. È evidente inoltre dalla matrice di confusione che l'algoritmo trova un'elevata quantità di falsi positivi.



Come si vede dal grafico, anche l'average precision è la più bassa rilevata: 0.31. Tuttavia la Precision si mantiene quasi costante rispetto al Recall.

Il risultato è tutto sommato prevedibile in quanto in questo algoritmo non si è usata la cross validation.

5. APPENDIMENTO: CONCLUSIONI

	Accuratezza	Varianza	Deviazione standard	F1-score	Average precision	AUC
Knn	0.75	0.001	0.034	0.44	0.49	0.688
Random forest	0.72	0.005	0.069	0.40	0.39	0.725
Multinomial naive Bayes	0.69	0.016	0.128	0.43	0.37	0.673
SVM	0.67	0.003	0.058	0.29	0.37	//
K-means	0.51	//	//	0.41	0.31	//

Dalla tabella riassuntiva si evince come, per il dataset utilizzato, l'accuratezza della classificazione migliore si ha nel Knn. I risultati poco soddisfacenti per gli altri algoritmi sono riconducibili allo sbilanciamento del dataset. Una possibile soluzione potrebbe essere quella di usare tecniche di ricampionamento più precise (non solo la cross-validation).

6. CSP

Molti problemi nell'ambito dell'Intelligenza Artificiale sono classificabili come Problemi di Soddisfacimento di Vincoli (Constraint Satisfaction Problem, CSP). Formalmente, un CSP può essere definito su un insieme finito di variabili (X_1, X_2, \dots, X_n) i cui valori appartengono a domini finiti di definizione (D_1, D_2, \dots, D_n) e su un insieme di vincoli (constraints) (C_1, C_2, \dots, C_n). Un vincolo su un insieme di variabili è una restrizione dei valori che le variabili possono assumere simultaneamente. Concettualmente, un vincolo può essere visto come un insieme che contiene tutti i valori che le variabili possono assumere contemporaneamente: un vincolo tra k variabili $C(X_{i1}, X_{i2}, \dots, X_{ik})$, è un sottoinsieme del prodotto cartesiano dei domini delle variabili coinvolte $D_{i1}, D_{i2}, \dots, D_{ik}$ che specifica quali valori delle variabili sono compatibili con le altre. Questo insieme può essere rappresentato in molti modi, per esempio per mezzo di matrici, equazioni, disuguaglianze o relazioni. Si definisce grado della variabile il numero di vincoli a cui è sottoposta. Un problema di soddisfacimento di vincoli presuppone un assegnamento iniziale, ovvero un insieme di variabili già vincolate. L'assegnamento iniziale può anche essere vuoto. La risoluzione del problema prosegue estendendo l'assegnamento iniziale, ovvero assegnando via via valori alle variabili ancora libere. La soluzione di un CSP è un assegnamento completo e coerente di valori alle variabili (ovvero un assegnamento che soddisfi tutti i vincoli e non lasci variabili libere), ottenuto estendendo l'assegnamento iniziale.

La libreria utilizzata (Constraint) ha permesso di realizzare un semplice CSP in grado di mostrare la disponibilità dei laboratori d'analisi nell'ipotetico caso di una visita istologica da parte di un paziente. Il CSP è relativo agli orari di apertura dei laboratori di analisi (per esempio: "Laboratorio A" fa le analisi dalle ore 8 alle 14). Il sistema indica i possibili orari a cui l'utente può prenotarsi.

```
In [36]: lab_booking()
Vuoi prenotare una visita istologica presso un laboratorio di analisi?
[si/no]

si
[0] Laboratorio Analisi istologiche A
[1] Laboratorio Analisi istologiche B
[2] Laboratorio Analisi istologiche C
[3] Laboratorio Analisi istologiche D

Seleziona il laboratorio: [0/1/2/3]

3
Disponibilita' laboratorio confermata.

Turno [0], Giorno: giovedi, Orario: 7
Turno [1], Giorno: giovedi, Orario: 8
Turno [2], Giorno: giovedi, Orario: 9
Turno [3], Giorno: giovedi, Orario: 10
Turno [4], Giorno: giovedi, Orario: 11
Turno [5], Giorno: sabato, Orario: 12
Turno [6], Giorno: giovedi, Orario: 12
Turno [7], Giorno: sabato, Orario: 13
Turno [8], Giorno: sabato, Orario: 14

Seleziona un turno inserendo il numero del turno associato:

8
Turno selezionato: [8], Giorno: sabato, Orario: 14
```

La libreria ha a disposizione tre solver: *Backtracking*, *Recursive backtracking*, *Minimum conflicts*. Il solver di default usato dalla libreria usa la tecnica del Backtracking. Data la semplice natura del problema in esame, si è deciso di utilizzare quest'ultimo.

6.1 Backtracking search

La tecnica backtracking-search è una ricerca depth-first che sceglie i valori per una variabile alla volta e torna indietro quando una variabile non ha più valori legali da assegnare. L'algoritmo di backtracking sceglie ripetutamente una variabile non assegnata, quindi prova a turno tutti i valori nel dominio di quella variabile, cercando di trovare una soluzione. Se viene rilevata un'incoerenza, BACKTRACK restituisce un errore, facendo sì che la chiamata precedente tenti un altro valore. Questo algoritmo mantiene solo una singola rappresentazione di uno stato e altera quella rappresentazione piuttosto che crearne di nuove.

```
function BACKTRACKING-SEARCH(csp) returns a solution, or failure
  return BACKTRACK({ }, csp)

function BACKTRACK(assignment, csp) returns a solution, or failure
  if assignment is complete then return assignment
  var ← SELECT-UNASSIGNED-VARIABLE(csp)
  for each value in ORDER-DOMAIN-VALUES(var, assignment, csp) do
    if value is consistent with assignment then
      add {var = value} to assignment
      inferences ← INFERENCE(csp, var, value)
      if inferences ≠ failure then
        add inferences to assignment
        result ← BACKTRACK(assignment, csp)
        if result ≠ failure then
          return result
      remove {var = value} and inferences from assignment
  return failure
```