Mushrooms Classification

Raffaele Anselmo[1], Lorenzo Pastore[2]

Sommario
La pratica del "mushroom hunting" è diffusa in gran parte di Europa, Australia, Giappone e Corea, in alcune zone del Medio Oriente e del subcontinente Indiano, così come nelle regioni temperate di Canada e Stati Uniti. [1] Sebbene sia un'attività ad alto rischio, negli ultimi anni sta godendo nuovi picchi di popolarità. L'analisi in questo report ha l'obiettivo di mettere in luce il potenziale delle tecniche di machine learning nell'ambito della classificazione. In particolare si vuole individuare il modello che meglio classifica i funghi in "velenosi" e "commestibili". Per riuscire nell'obiettivo sono state utilizzate diverse tecniche di validazione e sono stati applicati, testati e discussi alcuni tra i più noti modelli di classificazione.

Keywords
Machine Learning — Classification — Mushrooms — Data Science

[1] *Università degli studi di Milano-Bicocca, CdLM DataScience*
[2] *Università degli studi di Milano-Bicocca, CdLM DataScience*

Indice

Introduzione

I funghi selvatici che crescono nelle foreste e nei prati, sono di vario tipo, ed è spesso comune per la popolazione locale consumarli. L'esperienza e l'osservazione consentono ad un occhio esperto di discriminare i funghi tra velenosi e non velenosi, tuttavia chi non ha familiarità con i funghi può facilmente confonderne o mal interpretarne le caratteristiche. Di fatti il regno dei funghi comprende più di 100.000 specie conosciute, sebbene la diversità sia stimata in più di 3 milioni di specie. A seconda della specie, il consumo di funghi velenosi può causare vari sintomi, che vanno da allucinazioni, lievi disturbi, principalmente gastrointestinali, ad insufficienza epatica e morte. [2]

Il consumo di funghi ed il numero di casi di intossicazioni varia a seconda del paese. Tuttavia, nonostante la diffusa e longeva tradizione di questa pratica, sia il numero di specie classificate come commestibili che il numero di specie classificate come velenose risulta irisorio rispetto alla totalità delle specie esistenti.

L'obiettivo di questa indagine è di cercare di classificare i funghi in commestibili o velenosi a seconda delle loro caratteristiche al fine di capire se alcune tra le tecniche di *Machine Learning* più conosciute possono rivelarsi utili in questo tipo di classificazione. Per eseguire l'analisi ci siamo basati su dati di campioni ipotetici corrispondenti a 23 specie di funghi lamellati nelle famiglie di Agaricus e Lepiota tratte dalla "The Audubon Society Field Guide to North American Mushrooms". Il report è suddiviso in 5 paragrafi:

- Nel primo paragrafo viene presentato il dataset insieme ad una descrizione delle singole variabili;
- Nel secondo paragrafo sono raccolte le analisi preliminari sul dataset che hanno portato alla definizione delle opportune tecniche di missing replacement, partitioning, binarizzazione e data preparation;
- Nel terzo paragrafo vengono presentati i modelli di classificazione utilizzati;
- Nel quarto paragrafo i modelli ottenuti sono stati messi a confronto grazie ad alcune misure della valutazione delle performance e tramite l'utilizzo della ROC curve;

Infine, nell'ultima sezione, sono stati commentati i risultati ottenuti nell'analisi e sono stati messi in evidenza gli aspetti più importanti emersi dall'indagine.

1. Dataset

Per l'analisi è stato utilizzato il dataset "Mushrooms Classification", reso disponibile sulla piattaforma Kaggle [3] dall'UCI Machine Learning Repository. Questo dataset contiene le descrizioni di un campione ipotetico di 8124 funghi. Il dataset contiene 22 variabili:

1. **cap-shape:** forma del cappello

   bell = b, conical = c, convex = x, flat = f, knobbed = k, sunken = s

2. **cap-surface:** superficie del cappello

   fibrous = f, grooves = g, scaly = y, smooth = s

3. **cap-color:** colore del cappello

   brown = n, buff = b, cinnamon = c, gray = g, green = r, pink = p, purple = u, red = e, white = w, yellow = y

4. **bruises:** presenza di macchie

   bruises = t, no = f

5. **odor:** odore emanato

   almond = a, anise = l, creosote = c, fishy = y, foul = f, musty = m, none = n, pungent = p, spicy = s

6. **gill-attachment:** congiuzione delle lamelle

   attached = a, descending = d, free = f, notched = n

7. **gill-spacing:** spaziatura tra le lamelle

   close = c, crowded = w, distant = d

8. **gill-size:** dimensione delle lamelle

   broad = b, narrow = n

9. **gill-color:** colore delle lamelle

   black = k, brown = n, buff = b, chocolate = h, gray = g, green = r, orange = o, pink = p, purple = u, red = e, white = w, yellow =y

10. **stalk-shape:** forma dello stelo

    enlarging = e, tapering = t

11. **stalk-root:** radice dello stelo

    bulbous = b, club = c, cup = u, equal = e, rhizomorphs = z, rooted = r, missing = ?

12. **stalk-surface-above-ring:** superfice dello stelo al di sopra dell'anello

    fibrous = f, scaly = y, silky = k, smooth = s

13. **stalk-surface-below-ring:** superfice dello stelo al di sotto dell'anello

    fibrous = f, scaly = y, silky = k, smooth = s

14. **stalk-color-above-ring:** colore dello stelo al di sopra dell'anello

    brown = n, buff = b, cinnamon = c, gray = g, orange = o, pink = p, red = e, white = w, yellow = y

15. **stalk-color-below-ring:** colore dello stelo al di sotto dell'anello

    brown = n, buff = b, cinnamon = c, gray = g, orange = o, pink = p, red = e, white = w, yellow = y

16. **veil-type:** tipologia di velo

    partial = p, universal = u

17. **veil-color:** colore del velo

    brown = n, orange = o, white = w, yellow = y

18. **ring-number:** numero di anelli

    none=n,one=o,two=t

19. **ring-type:** tipologia di anello

    cobwebby = c, evanescent = e, flaring = f, large = l, none = n, pendant = p, sheathing = s, zone = z

20. **spore-print-color:** colore dell'impronta sporale

    black = k, brown = n, buff = b, chocolate = h, green = r, orange = o, purple = u, white = w, yellow = y

21. **population:** popolazione

    abundant = a, clustered = c, numerous = n, scattered = s, several = v, solitary = y

22. **habitat:** tipologia di habitat

    grasses = g, leaves = l, meadows = m, paths = p, urban = u, waste = w, woods = d

Si nota inoltre che il dataset "Mushrooms Classification" è caratterizzato da **classi bilanciate**. Nell'ambito della nostra analisi la classe positiva (poisonous=p) è pari a 3916 su 8124, ovvero al 48,2% del totale.

2. Preprocessing

Delle 22 variabili iniziali è stata omessa una variabile poichè assumeva un unico valore per tutti i records ed e stata individuata la variabile "stalk-root" a causa della forte presenza di outlier.

Nello specifico sono state effettuate le seguenti operazioni:

- *veil-type (tipologia di velo):* rimossa poichè assumeva il valore "partial" per tutte le osservazioni
- *stalk-root (radice dello stelo):* circa 31% di missing values sono stati sostituiti con la tecnica del "mode replacement"

A tal proposito si precisa che le analisi svolte fanno riferimento ad dataset di funghi che condividono tutti la stessa tipologia di velo.

Dopo aver effettuato le operazioni di rimozione e missing replacement, i rimanenti attributi, essendo tutti nominali, sono stati binarizzati.

2.1 Binarization

Attraverso la binarizzazione, in generale, si rendono "metriche" le variabili qualitative, trasformando ogni variabile qualitativa in tante variabili binarie quante sono le modalità della stessa. Nel caso specifico le variabili dicotomiche *bruises*, *gill-attachment*, *gill-spacing*, *gill-size* e *stalk-shape*, sono state separate dalle restanti e trasformate in variabili dummy, mentre le rimanenti sono state binarizzate in accordo con il numero di valori assunti da ciascun attributo.

2.2 Principal Component Analysis

La PCA è una tecnica di algebra lineare che consente di proiettare i dati da un *"high-dimensional space"* in un *"lower-dimensional space"*. Questa tecnica individua una serie di componenti dette principali a partire da combinazioni lineari degli attributi originali che sono tra di loro ortogonali e consentono di catturare la massima quantità di variabilità nei dati. I vantaggi della riduzione del numero di attributi sono dati sia da una migliore performance degli algoritmi che da una maggiore interpretabilità del modello sviluppato. In questa analisi si è deciso di applicare la PCA agli attributi binarizzati [4], in particolare mantenendo il 90% della variabilità presente all'interno dei dati si utilizzano 32 componenti principali.

2.3 Partitioning

Nel corso dello sviluppo dell'analisi, sono state applicate due tecniche di partizionamento dei dati.

Nella prima partizione si è deciso di utilizzare il metodo *hold-out*, con il quale il dataset originale è stato suddiviso in due subset, il 67% dei dati è stato attribuito alla partizione A (training set) ed il restante 33% alla partizione B (test set). Inoltre essendo la class attribute già bilanciata si è ritenuto opportuno utilizzare un campionamento casuale.

La seconda tecnica di partizionamento è stata applicata durante la fase di apprendimento dei modelli. La K-fold cross validation, che consiste nella divisione del dataset in K parti di uguale numerosità, è stata applicata ai learner dei diversi modelli durante la fase di apprendimento.

3. Data Modelling

Il primo passo è stato decidere una serie di algoritmi di Machine Learning tenendo in considerazione sia la struttura dei dati che il l'obiettivo dell'analisi.

Gli attributi sono tutti nominali categorici (di cui una parte dicotomica), mentre la "class attribute" è una variabile dicotomica che assume solo i valori "e = edible" e "p = poisoning". Nello specifico i modelli di classificazione utilizzati fanno riferimento a cinque distinte categorie:

- **Modelli probabilistici:** si basano sulla teoria Bayesiana e consentono in linea generale di ottenere risultati accurati. Tuttavia gli attributi che descrivono le istanze sono condizionalmente indipendenti data la classificazione (anche se spesso questa ipotesi può essere violata). Tra questi è stato implementato il classificatore *Naïve Bayes*.

- **Modelli euristici:** nonostante non garantiscano di giungere a risultati ottimali, sono in grado di ottenere soluzioni approssimate ragionevoli senza richiedere sforzi computazionali eccessivi o ipotesi restrittive sui dati di input. Questi metodi fanno riferimento in particolar modo agli alberi decisionali , tra questi modelli si è concentrata l'attenzione sul classificatore *Decision Tree* di Knime. In accordo con i nodi per la parameter optimization abbiamo settato il numero minimo di osservazioni in una foglia pari a 2.

- **Modelli di separazione:** tentano di mappare i i dati in uno spazio di dimensioni superiore, con l'obiettivo di trovare l'iperpiano che meglio separi le variabili in base alla variabile di interesse utilizzando delle *kernel functions*. Tra questi è stata integrata una Support Vector Machine (SVM) utilizzando il nodo *SMO(3.7)* di Weka.

- **Modelli di regressione:** risultano molto flessibili e facilmente comprensibili in quanto è possibile misurare l'effetto delle diverse variabili nella classificazione grazie ai coefficienti assegnati dal modello. La regressione logistica è un caso particolare di modello lineare generalizzato avente come funzione link la funzione logit, Si tratta di un modello di regressione applicato nei casi in cui la variabile dipendente sia di tipo dicotomico. Nell'analisi è stato implementato il nodo *Logistic Regression* di Knime. Inoltre sono stati utilizzati i nodi *loop parameter optimization* per trovare i valori di step-size e varianza che massimizzano l'Area sottostante la ROC curve.

- **Modelli di reti neurali:** sono modelli di tipo *black-box* che sfruttano una rete di neuroni artificiali per stimare funzioni complesse. Tuttavia proprio a causa della loro struttura hanno il difetto di essere difficilmente interpretabili. Tra questi si è deciso di utilizzare un *Multilayer Perceptron*.

4. Models evaluation

In questa sezione vengono presentate le procedure che sono state effettuate per la valutazione dei modelli, le misure utilizzate per il confronto tra questi ed infine i risultati vengono presentati grazie ad alcune note tecniche grafiche.

4.1 Hold-out e Cross-Validation

Dopo aver effettuato le operazioni di preprocessing il dataset è stato suddiviso con il metodo *hold-out* in due partizioni, pari

rispettivamente al 67%, training set, e al 33%, test set, del dataset iniziale. Grazie al training set sono stati addestrati i modelli di classificazione selezionati, il cui confronto è stato successivamente possibile attraverso della valutazione delle loro performance sul test set.

Una seconda partizione è stata effettuata per ottenere una validazione dei classificatori utilizzando un approccio chiamato K-fold cross validation. Questa tecnica suddivide il dataset in K partizioni di eguale numerosità e assicura che tutti i record vengano utilizzati almeno una volta sia nel training set che nel validation set. Nel dividere il dataset in K partizioni di eguale ampiezza, si è scelto di utilizzare ancora una volta il campionamento casuale. Ad ogni passo, K-1 partizioni entrano a far parte del training set, su cui l'algoritmo di classificazione viene allenato, mentre la rimanente k-esima partizione è utilizzata come validation set, blocco di osservazioni di cui il modello predice il valore dell'attributo di classe. Per definire quale fosse il valore ottimale di iterazioni è stato inizialmente utilizzato il nodo *parameter optimization loop*. Tuttavia, questo processo è risultato computazionalmente troppo dispendioso, si è deciso pertanto di imputare il numero di iterazioni della Cross Validation a K=5.

4.2 Misure di valutazione della performance

La selezione delle misure di valutazione è stata effettuata a partire dalla struttura del problema e tenendo sempre in considerazione l'obiettivo di questa particolare applicazione, cioè effettuare una classificazione dei funghi in base alle loro caratteristiche cercando di ridurre al minimo la possibilità di incorrere in un fungo velenoso. Per tale motivo, delle possibili misure derivabili dalla confusion matrix che generalmente vengono utilizzate come indici di bonta' del modello, in particolare:

- *Precision:* $\frac{TP}{TP+FP}$
  Frazione di osservazioni effettivamente positive nel gruppo della classe prevista positiva
- *Recall:* $\frac{TP}{TP+FN}$
  Frazione di osservazioni positive correttamente previste dal modello
- *Specificity:* $\frac{TN}{TN+FP}$
  Frazione di osservazioni negative correttamente previste dal modello
- *F-measure:* $\frac{2*R*P}{R+P}$
  Media armonica tra Recall e Precision. Un valore alto ci indica che sia Recall che Precision assumono valori alti
- *Accuracy:* $\frac{TP+TN}{TP+FP+TN+FN}$
  Percentuale di osservazioni positive e negative predette correttamente

Sono state prese in considerazione le sole misure in grado di dare una migliore e più completa valutazione della performance.

4.3 Models Comparison

Al fine di individuare il classificatore migliore, sono state calcolate le quantità sovracitate relativamente ai risultati delle performance di ogni modello. Le misure sono riassunte nella tabella in Figura 1.

Classifiers	Recall	Precision	Specificity	F-measure	Accuracy
Naïve Bayes	0,804	0,997	0,998	0,89	0,903
Decision Tree	0,988	0,996	0,996	0,992	0,992
Support Vector Machine	0,999	1	1	1	1
Logistic Regression	0,97	0,988	0,989	0,979	0,98
Multilayer Perceptron	1	1	1	1	1

Figura 1. Valori di Accuracy, F-measure e Precision dei modelli di classificazione

Per ottenere una rappresenzatione grafica più esaustiva, i risultati sono stati visualizzati per mezzo di alcuni boxplot. La Figura 2 mostra i Boxplot dell'accuracy delle performance di ogni classificatore sul test set. I risultati migliori vengono ottenuti dai classificatori *Support Vector Machine* e *Multilayer Perceptron* entrambi con un valore pari ad 1.

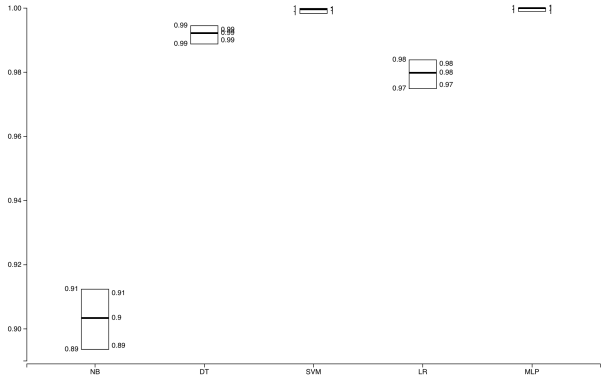

Figura 2. Boxplot dell'Accuracy dei modelli di classificazione

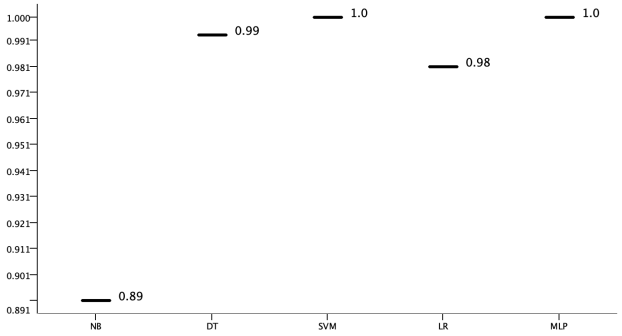

Figura 3. Boxplot dell'F-measure dei modelli di classificazione

In seguito si è voluto utilizzare una metrica che tenesse conto sia dei valori della *precision* che della *recall*. Pertanto sono stati confrontati i valori dell'*F-measure* ottenuti dai diversi classificatori (vedi Fig.3).

Dall'analisi dell'*F-measure* emergono risultati concordi a quelli evidenziati in precedenza dall'analisi dell'*accuracy*. Di fatti anche in questo caso i valori più elevati sono raggiunti da *Support Vector Machine* e *Multilayer Perceptron* ad indicare livelli molto elevati di *precision* e *recall*.

Infine per svolgere un'analisi grafica più approfondita, si è deciso di confrontare le performance dei classificatori anche per mezzo delle relative ROC curve.

4.4 ROC Curve

Una tecnica grafica nota per confrontare modelli di classificatori è la ROC curve. Questa viene rappresentata considerando tutti i possibili valori del test e, per ognuno di questi, viene calcolata la proporzione di *true-positive* e quella di *false-positive*. Generalmente nel caso di un classificatore perfetto, l'area sottostante la curva di ROC è pari a 1 mentre, se un modello ha basse capacità predittive, è inferiore a 0,5.

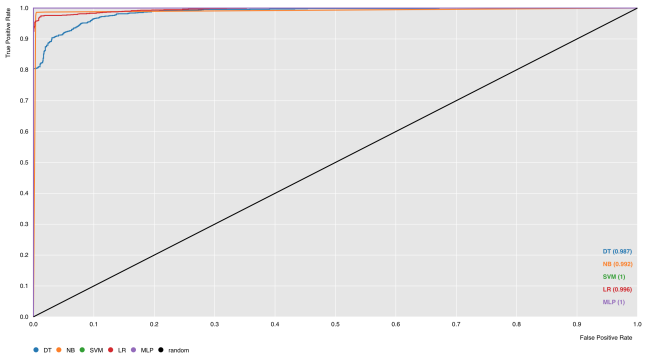

Figura 4. ROC Curve dei modelli di classificazione

Nella figura 4 sono rappresentate le curve riferite ai modelli analizzati nella sezione precedente. Il valore dell'AUC, cioè l'area under the curve, ci fornisce un'ulteriore una misura di accuratezza del modello. Anche in questo caso, in maniera del tutto coerente con i risultati ottenuti nella sezione 4.3, emergono i risultati della *SVM* (1) e del *Multilayer Perceptron* (1), seguiti dalla *Logistic Regression* (0,996) e dal *Naive Bayes* (0,992).

5. Conclusioni

Lo scopo di questo elaborato è di mettere in luce il potenziale delle tecniche di machine learning. Nello specifico è stato affrontato un problema di classificazione utilizzando cinque diversi modelli differenti. L'obiettivo primario è stato trovare il modello di classificazione migliore tra quelli proposti e testati.

Il problema principale che si è tentato di risolvere è stato quello della riduzione della dimensionalità, a seguito delle operazioni di binarizzazione, per ottenere performance migliori dagli algoritmi di classificazione. In particolare è stata effettuata una PCA che ha consentito di ridurre il numero da 110 attributi a 32 componenti principali, mantenendo il 90% della variabilità dei dati.

Dall'analisi emergono in maniera evidente le performance di due classificatori la *Support vector machine* ed il *Multilayer perceptron*. Tuttavia, considerando particolarmente significativo il richio in cui si potrebbe incorrere nel caso di un fungo velenoso erroneamente classificato, abbiamo deciso di confrontare i modelli anche secondo i valori della *Recall*.

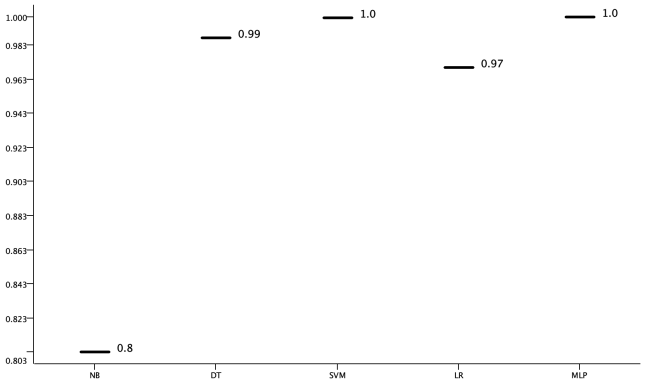

Figura 5. Boxplot della Recall dei modelli di classificazione

I boxplot della *recall* confermano i risultati ottenuti in precedenza, ed evidenziano per il modello *Naive Bayes* una maggiore difficoltà nel classificare correttamente i funghi velenosi.

In conclusione si evince come i migliori classificatori siano la *Support vector machine* ed il *Multilayer perceptron*, sebbene buone performance siano state ottenute anche dalla *Logistic Regression*, utile ai fini inferenziali.

In questo elaborato è stata utilizzata una PCA, tuttavia possibili sviluppi futuri potrebbero essere affrontati considerando altre tecniche o variazioni della stessa, quale la categorical PCA, non disponibile su Knime. Un'ulteriore sviluppo potrebbe invece escludere la PCA al fine di capire quali sono le caratteristiche maggiormente indicative dei funghi velenosi attraverso l'interpretazione dei coefficienti della regressione logistica.

Riferimenti bibliografici

[1] https://en.wikipedia.org/wiki/Mushroom_hunting.

[2] https://en.wikipedia.org/wiki/Mushroom_poisoning.

[3] Kaggle:https://www.kaggle.com/uciml/mushroom-classification.

[4] J.C.Gower. Some distance properties of latent root and vector methods used in multivariate analysis. *Biometrika*, 53:325–338, 1966.