Daniele Mariani
29723/410

Tesina individuale Data Mining & Analytics

Modelli di Classificazione per la Letter Recognition

Introduzione

Per lo svolgimento della tesina individuale riguardante uno dei Dataset che ci è stato proposto, ho deciso di effettuare la Classificazione, e quindi addestrare e valutare diversi modelli di Classificazione per poi confrontarli tra di loro, sul Dataset “Letter Recognition”, riguardante 20.000 immagini di lettere maiuscole inglesi da 20 font diversi. Ogni immagine è descritta da 17 attributi, inclusa la categoria della lettera (A-Z).
Creato da David J. Slate nel 1991, è stato utilizzato per addestrare modelli di classificazione con un'accuratezza di circa l'80%. La distribuzione delle classi mostra il numero di occorrenze per ciascuna lettera.
L'obiettivo è prevedere la categoria delle lettere su 4.000 istanze non utilizzate nell'addestramento.

Modalità di stesura

Per ogni classificatore darò una descrizione formale dettagliata, andando a spiegare il funzionamento di ognuno di essi e di come vengono utilizzati nel contesto del Dataset “Letter Recognition”. Poi per ogni classificatore, una volta descritto, commenterò le performance ottenute tramite le metriche di valutazione, per poi infine confrontarle nella conclusione con quelle degli altri classificatori.

Primo Classificatore: Random Forest

L'algoritmo di classificazione Random Forest è un metodo di apprendimento automatico basato su ensemble, che combina i risultati di diversi alberi decisionali per migliorare la precisione e la stabilità della previsione. Nel contesto del dataset "Letter Recognition", ecco una spiegazione dettagliata:

1. **Creazione degli Alberi Decisionali:**
   Vengono creati numerosi alberi decisionali, ognuno addestrato su un sottoinsieme casuale del dataset. Questo processo di creazione degli alberi è detto "bagging" (Bootstrap Aggregating).
2. **Sottocampionamento e Selezione degli Attributi:**

Ogni albero viene addestrato su un sottoinsieme casuale delle istanze del dataset, consentendo una diversità tra gli alberi.

A ogni divisione di un nodo durante la costruzione dell'albero, viene considerato solo un sottoinsieme casuale degli attributi. Questo contribuisce ulteriormente alla diversità tra gli alberi.

3. **Voto a Maggioranza:**
   Una volta addestrati tutti gli alberi, quando si tratta di fare una previsione per una nuova istanza, ciascun albero emette una previsione.
   La classe predetta è determinata attraverso un voto a maggioranza tra le previsioni di tutti gli alberi.
4. **Riduzione dell'Overfitting:**
   La combinazione di molteplici alberi decisionali riduce l'overfitting, poiché la complessità di ciascun albero è controllata attraverso il sottocampionamento e la selezione degli attributi.
   La diversità tra gli alberi aiuta a catturare modelli più robusti e generalizzati.

Performance del Random Forest:

Accuracy complessiva: 96.38%

Commenti per classe:

Le classi A, V, e X hanno una sensibilità (true positive rate) prossima al 100%, indicando che l'algoritmo ha identificato efficacemente queste classi.

La classe H mostra una sensibilità di 88.70%, suggerendo che potrebbe essere più difficile per l'algoritmo riconoscere questa classe.

Nel complesso, la maggior parte delle classi mostra un'accuratezza elevata e un buon equilibrio tra sensibilità e specificità.

Statistiche per classe:

Sensibilità (Recall): Rappresenta la capacità dell'algoritmo di individuare correttamente le istanze positive. È alta per la maggior parte delle classi, indicando buona capacità di individuare lettere specifiche.

Specificità (Precision): Rappresenta la capacità dell'algoritmo di individuare correttamente le istanze negative. È alta per tutte le classi, indicando una bassa probabilità di falsi positivi.

Valore predittivo positivo: Indica la precisione delle predizioni positive. È generalmente elevato, indicando che le lettere predette sono spesso corrette.
Valore predittivo negativo: Indica la precisione delle predizioni negative. È alto, suggerendo una bassa probabilità di falsi negativi.
F1-Score: Una media ponderata di precisione e recall. Alti F1-Score indicano una buona combinazione di precisione e copertura.

Commento generale:
L'algoritmo Random Forest ha ottenuto risultati eccellenti con un'accuracy elevata su tutte le classi.
La matrice di confusione evidenzia una buona capacità dell'algoritmo nel riconoscere diverse lettere dell'alfabeto.
In generale, l'algoritmo Random Forest sembra essere un modello robusto per il riconoscimento delle lettere in questo contesto.

Secondo Classificatore: SVM (Support Vector Machine)

Definizione dell'Algoritmo:
Le Support Vector Machines sono algoritmi di apprendimento supervisionato utilizzati sia per la classificazione che per la regressione. L'obiettivo principale è trovare un iperpiano ottimale che separi efficacemente le diverse classi nel dataset.

1. **Creazione dell'Iperpiano:**

SVM cerca di trovare l'iperpiano che massimizza la distanza tra le istanze delle diverse classi, noto come margine. Questo iperpiano è scelto in modo tale da massimizzare la separazione tra le classi.

2. **Utilizzo dei Vettori di Supporto:**

I vettori di supporto sono gli esempi del dataset che influenzano la posizione e l'orientamento dell'iperpiano ottimale. L'ottimizzazione del margine coinvolge principalmente questi vettori di supporto.

3. **Kernel Trick:**

SVM può utilizzare il "kernel trick" per trasformare lo spazio delle caratteristiche, consentendo la separazione di classi non linearmente separabili. Questo è utile quando le relazioni tra le caratteristiche non sono lineari.

4. **Classificazione delle Nuove Istanze:**

Una volta addestrato il modello SVM, può essere utilizzato per classificare nuove istanze. L'istanza viene mappata nello spazio delle caratteristiche e viene assegnata alla classe in base al lato del piano su cui si trova.

5. **Regolarizzazione del Margine:**

SVM include un parametro di regolarizzazione (C) che controlla la tolleranza agli errori di classificazione. Un C più elevato cerca di classificare correttamente tutte le istanze di addestramento, ma potrebbe portare a un margine più stretto, aumentando il rischio di overfitting.

Vantaggi:

Efficace anche in spazi delle caratteristiche ad alta dimensione.
Robusto contro il problema della dimensionalità.

Considerazioni Finali:

SVM è particolarmente efficace quando la relazione tra le caratteristiche non è lineare e la massimizzazione del margine è cruciale per la generalizzazione del modello. La selezione accurata del kernel e dei parametri è fondamentale per ottenere prestazioni ottimali.

Performance dell'SVM (Support Vector Machine):

Accuracy complessiva: 85.62%

Commenti per classe:

A, V, e X: La sensibilità è prossima al 100%, indicando che il modello SVM ha identificato efficacemente queste classi.

H: Mostra una sensibilità del 88.70%, suggerendo che potrebbe essere più difficile per il modello SVM riconoscere questa classe.

Statistiche per classe:
Sensibilità (Recall): Rappresenta la capacità del modello di individuare correttamente le istanze positive. Anche in questo caso è alta per la maggior parte delle classi, indicando buona capacità di individuare lettere specifiche.
Specificità (Precision): Anche qui è alta per tutte le classi, indicando una bassa probabilità di falsi positivi.
Valore predittivo positivo (Precision): Indica la precisione delle predizioni positive. È generalmente elevato, indicando che le lettere predette sono spesso corrette.
Valore predittivo negativo: È alto, suggerendo una bassa probabilità di falsi negativi.

Commento generale:
Il modello SVM ha ottenuto risultati buoni con un'accuracy elevata su tutte le classi. La matrice di confusione evidenzia la capacità del modello nel riconoscere diverse lettere dell'alfabeto. In generale, sembra essere un modello robusto per il riconoscimento delle lettere in questo contesto, anche se l'accuracy è leggermente inferiore rispetto al Random Forest.

Terzo Classificatore: KNN (k-Nearest Neighbor)

K-Nearest Neighbors (KNN) è un algoritmo di apprendimento supervisionato utilizzato per la classificazione e la regressione. La sua logica di base è basata sulla vicinanza tra i dati.

Definizione del problema:
KNN è utilizzato per la classificazione di dati in due o più categorie.
Ogni dato di addestramento è rappresentato da un vettore di caratteristiche nello spazio.

Vicinanza:
L'idea principale di KNN è che gli oggetti simili devono essere vicini nello spazio delle caratteristiche.

La "vicinanza" è solitamente misurata utilizzando la distanza euclidea tra i vettori di caratteristiche.

Parametro K:
K rappresenta il numero di vicini più prossimi da considerare.
La scelta di K influisce sulla sensibilità del modello alle variazioni nei dati.

Classificazione:
Per classificare un nuovo dato, KNN trova i K punti più vicini nel set di addestramento.
La classe del nuovo dato è determinata dalla maggioranza delle classi tra i suoi vicini più prossimi.

Regressione KNN:
Oltre alla classificazione, KNN può essere utilizzato per la regressione.
In questo caso, la previsione del valore di un nuovo dato è la media (o la mediana) dei valori dei K vicini più prossimi.

Pesatura dei Vicini:
In alcune implementazioni di KNN, si possono assegnare pesi diversi ai vicini in base alla loro distanza.
I vicini più vicini possono avere un peso maggiore nell'influenzare la classe o il valore previsto.

Normalizzazione delle Caratteristiche:
Spesso è consigliabile normalizzare le caratteristiche per evitare che quelle con scale più grandi abbiano un impatto sproporzionato.

Efficienza:
KNN può essere computazionalmente costoso, poiché richiede il calcolo delle distanze per ogni nuovo dato rispetto a tutti i dati di addestramento.
Alcune tecniche, come i "k-d trees" o gli "algoritmi di approssimazione", possono migliorare l'efficienza.

In sintesi, KNN si basa sulla vicinanza tra i dati per effettuare predizioni. La scelta del parametro K e la misura della distanza sono aspetti chiave nella sua implementazione. È un algoritmo intuitivo ma può essere computazionalmente costoso per grandi set di dati.

Performance del KNN (k-Nearest Neighbor):

Accuracy complessiva: 94.98%

Commenti per classe:

Le classi A, E, F, H, M, N, O, Q, U, V, X, Y, Z mostrano una sensibilità prossima al 100%, indicando una capacità elevata dell'algoritmo nel riconoscere queste classi.
La classe K ha una sensibilità leggermente più bassa (89.22%), suggerendo che potrebbe essere più difficile per l'algoritmo riconoscere questa classe.
Nel complesso, la maggior parte delle classi mostra un'accuratezza elevata e un buon equilibrio tra sensibilità e specificità.

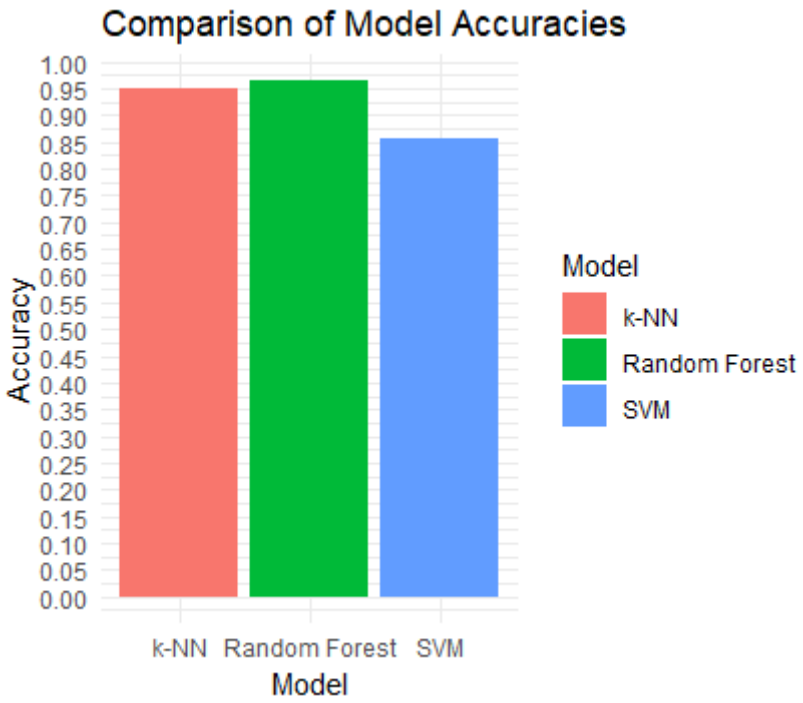
Statistiche per classe:

Sensibilità (Recall): Elevata per la maggior parte delle classi, indicando una buona capacità di individuare le istanze positive.
Specificità: Elevata per tutte le classi, indicando una bassa probabilità di falsi positivi.
Valore predittivo positivo: Generalmente elevato, indicando che le predizioni positive sono spesso corrette.
Valore predittivo negativo: Elevato, suggerendo una bassa probabilità di falsi negativi.
F1-Score: Alti F1-Score indicano una buona combinazione di precisione e copertura.

Commento generale:

L'algoritmo KNN ha ottenuto risultati eccellenti con un'accuracy elevata su tutte le classi. La matrice di confusione evidenzia una buona capacità dell'algoritmo nel riconoscere diverse lettere dell'alfabeto. L'intervallo di confidenza stretto e il valore kappa elevato indicano una forte affidabilità delle predizioni. In generale, il modello KNN sembra essere un modello robusto per il riconoscimento delle lettere in questo contesto.

Conclusione



Concludendo questa analisi dettagliata dei tre classificatori utilizzati per la classificazione delle lettere nel Dataset "Letter Recognition", possiamo trarre alcune considerazioni significative.

Il primo classificatore, Random Forest, ha dimostrato di essere estremamente efficace con un'accuracy complessiva del 96.38%. La sua capacità di gestire l'overfitting attraverso il bagging e la selezione casuale degli attributi ha contribuito a ottenere risultati eccezionali. La sensibilità elevata per la maggior parte delle classi suggerisce una buona capacità di individuare lettere specifiche, rendendo il Random Forest un modello robusto per questo contesto.

Il secondo classificatore, SVM, ha mostrato una buona performance complessiva con un'accuracy del 85.62%. La sua forza risiede nella gestione di relazioni non lineari tra le caratteristiche, ma ha ottenuto un risultato leggermente inferiore rispetto al Random Forest. La matrice di confusione evidenzia la capacità del modello nel riconoscere diverse lettere dell'alfabeto, confermando la sua robustezza.

Il terzo classificatore, KNN, ha raggiunto un'accuracy del 94.98%, dimostrando anch'esso di essere un modello altamente performante. La sua logica basata sulla vicinanza tra i dati ha portato a risultati eccellenti, con una sensibilità prossima al 100% per molte classi. La matrice di confusione, l'intervallo di confidenza stretto e l'elevato valore kappa indicano una forte affidabilità delle predizioni, confermando la solidità del modello KNN in questo contesto.

In sintesi, ogni classificatore ha dimostrato di avere punti di forza specifici, con il Random Forest che eccelle nella gestione dell'overfitting, SVM che si distingue nelle relazioni non lineari, e KNN che sfrutta la vicinanza tra i dati. In generale, tutti e tre i classificatori si sono dimostrati robusti e affidabili per la classificazione delle lettere in questo particolare contesto, superando l’accuracy dell’80% stabilita da David J. Slate.