Progetto di Laboratorio di Reti e Sistemi Distribuiti

Davide Mento

Università degli Studi di Messina

21/06/2024

Università degli Studi di Messina

L'algoritmo KNN (K-Nearest Neighbors)

L'algoritmo KNN è un metodo di apprendimento supervisionato per classificazione e regressione. Si basa sul principio che oggetti simili tendono a trovarsi vicini nello spazio delle caratteristiche.

- Non richiede una fase di addestramento complessa.
- Per fare una previsione per un nuovo punto, trova i K punti più vicini nel set di addestramento e determina la classe basandosi sulla maggioranza delle classi dei suoi vicini.

Esempio KNN

L'algoritmo KNN determina la classe di un nuovo punto basandosi sulla maggioranza delle classi dei K punti più vicini nel set di addestramento.

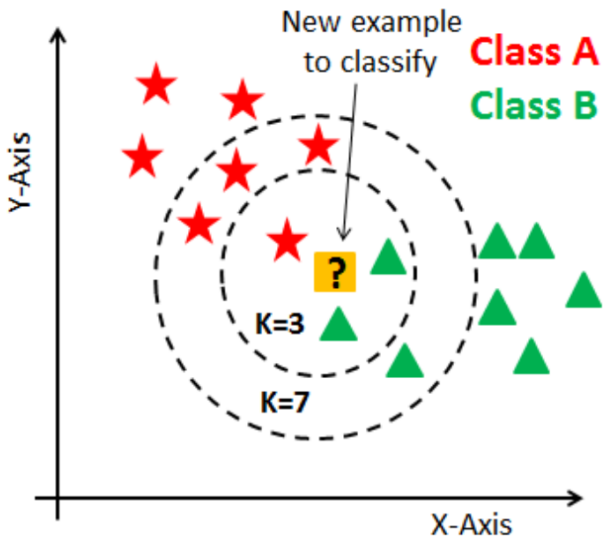
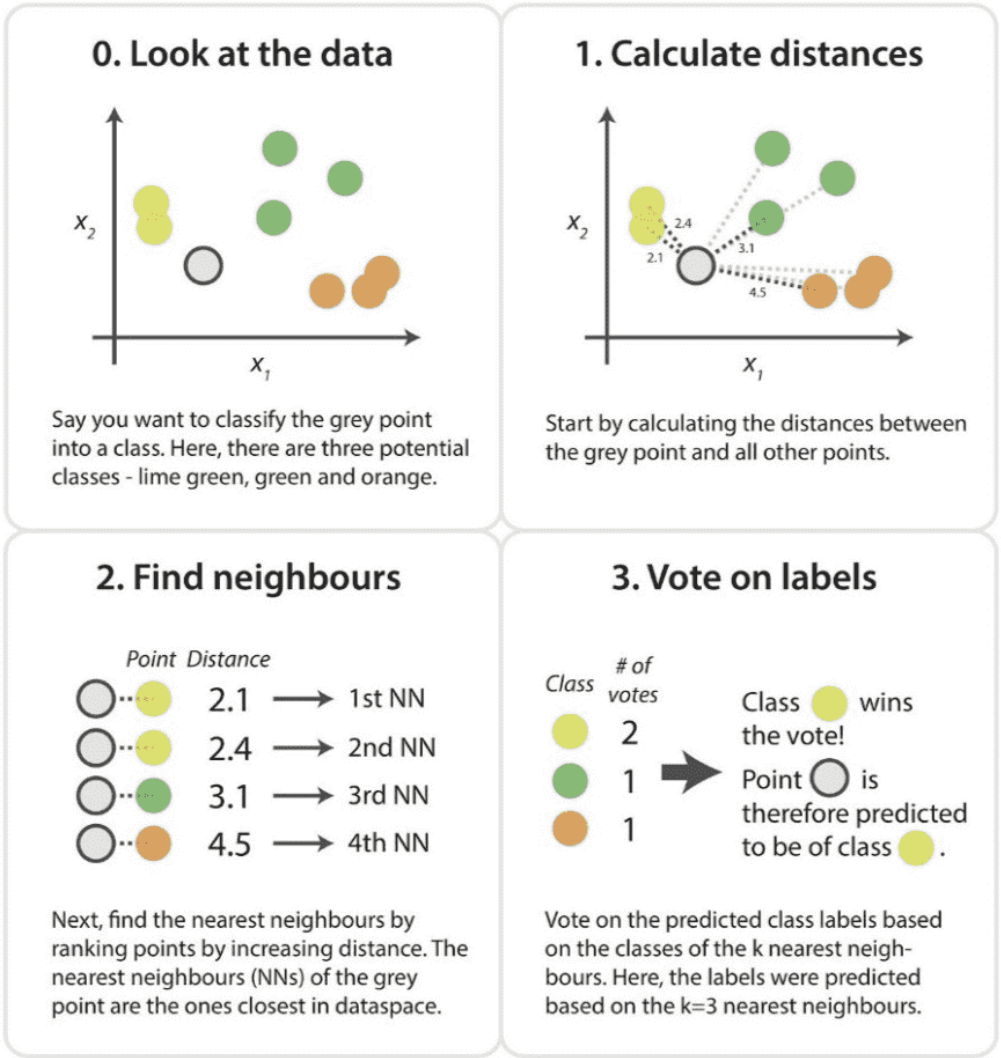


Diagramma KNN



Pregi e Difetti del KNN

Pregi:

- Semplicità concettuale e implementativa.
- Non richiede una fase di addestramento complessa.
- Buone prestazioni con dati di alta qualità e ben preprocessati.

Difetti:

- Prestazioni computazionali lente, soprattutto con grandi set di dati.
- Necessità di determinare un valore appropriato di K, che può influenzare significativamente le prestazioni.
- Memoria significativa richiesta per memorizzare l'intero set di addestramento e le distanze tra i punti.

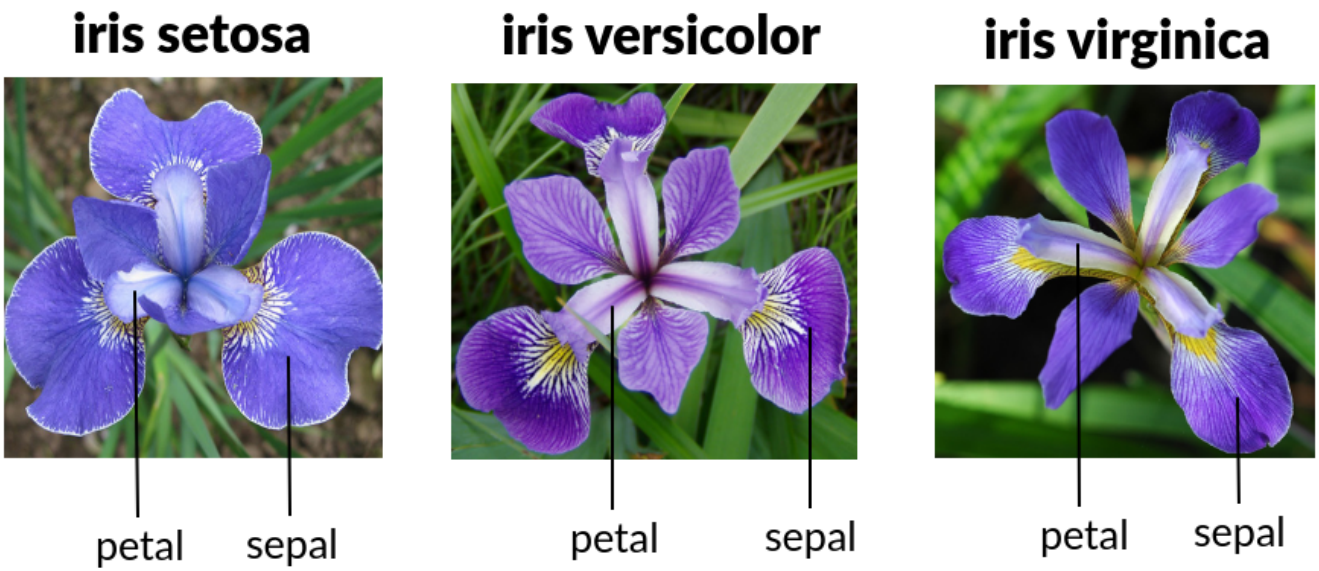
Efficienza del KNN

Data la pesantezza dell'algoritmo KNN, è necessario trovare una soluzione distribuita per migliorarne le prestazioni.

- KNN è computazionalmente intensivo, specialmente con grandi set di dati, a causa della sua natura basata sul calcolo delle distanze.
- L'approccio distribuito può suddividere il carico di lavoro e sfruttare risorse parallele per migliorare l'efficienza complessiva.
- Una soluzione distribuita potrebbe mitigare i problemi di memoria e velocizzare l'esecuzione dell'algoritmo su grandi volumi di dati.

Il Dataset Iris

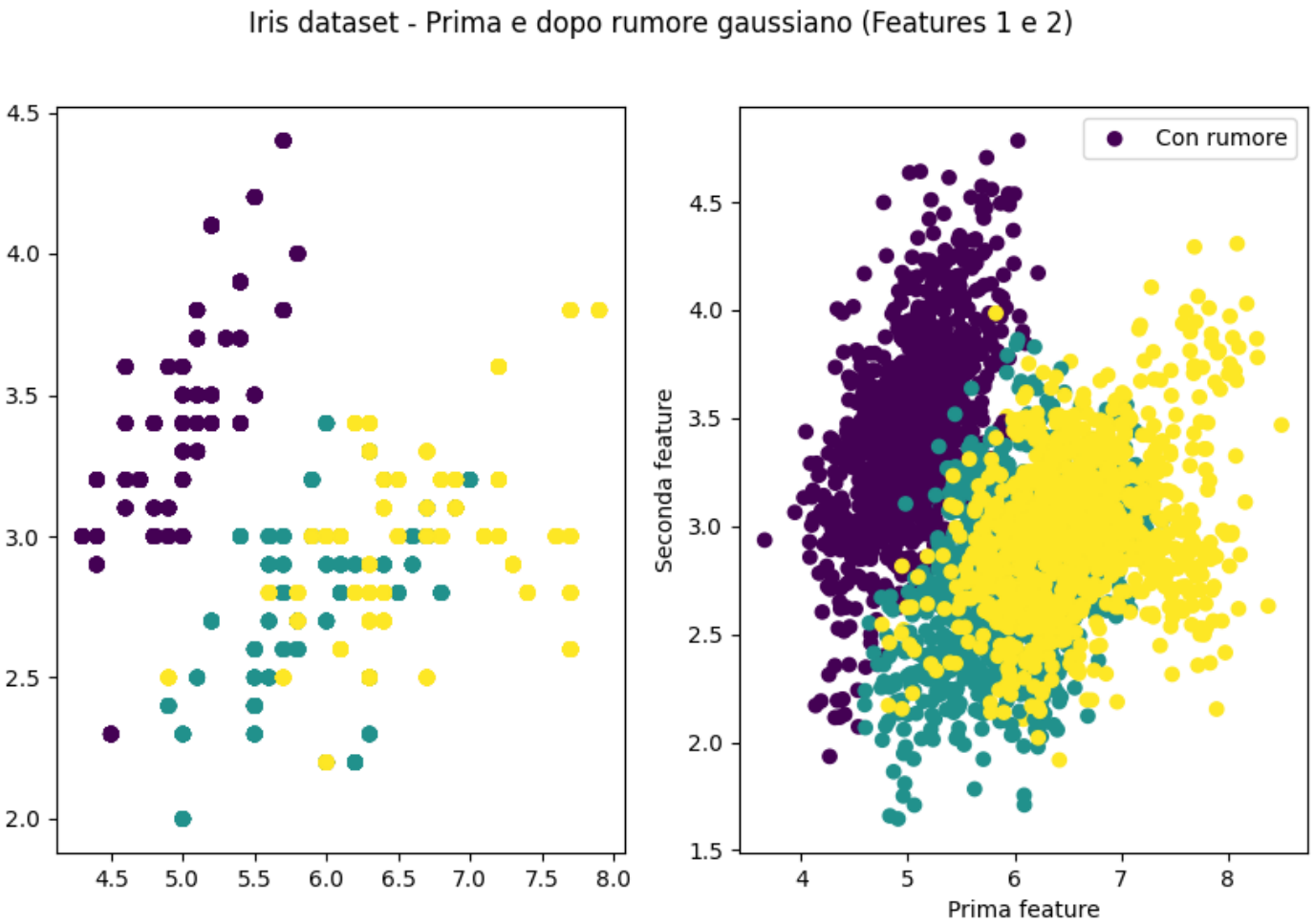
È stato scelto il dataset Iris per la sua popolarità nella comunità scientifica e per la sua divisione ben organizzata delle classi.



Data Augmentation e Aggiunta di Rumore

- **Data Augmentation**: Il dataset è stato ampliato tramite RandomOverSampling.
- **Aggiunta di Rumore**: È stato introdotto rumore gaussiano per simulare condizioni reali e migliorare il train set.

Visualizzazione dataset modificato



Analisi delle Componenti Principali (PCA)

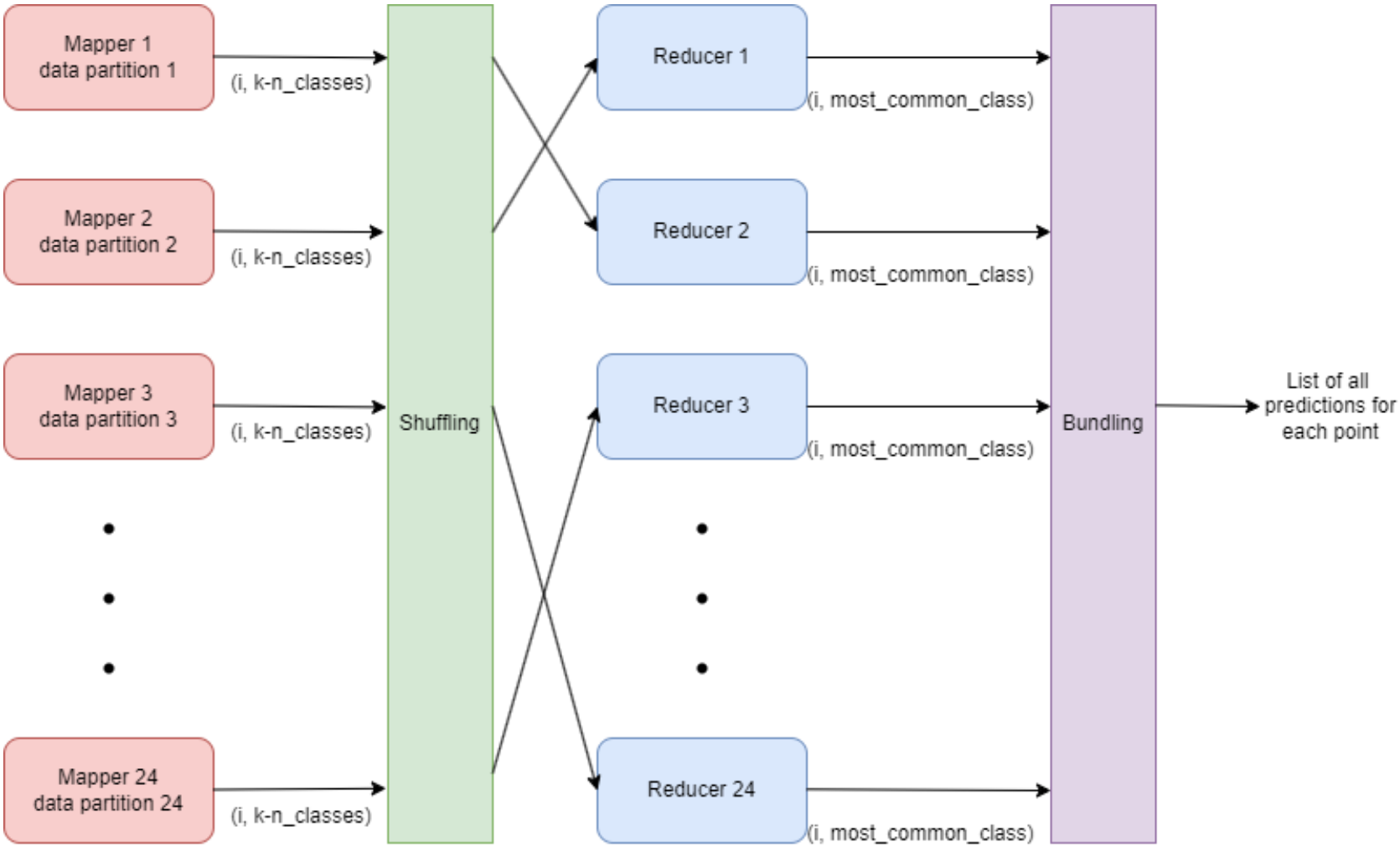
- **PCA**: È stata applicata la PCA (Principal Component Analysis) per ridurre la dimensionalità dei dati mantenendo le loro caratteristiche principali.
- **Benefici della PCA**: La PCA si è rivelata fondamentale nel rappresentare graficamente le relazioni tra le caratteristiche degli Iris.

MapReduce

Per gestire l'algoritmo KNN in modo efficiente su grandi dataset, è stato adottato un approccio basato su MapReduce utilizzando Ray.

- **Ray**: Framework per il calcolo distribuito.
- **Fase di Map**: Ogni mapper riceve una porzione di dati, su di questa viene fatto il calcolo delle distanze e delle K classi più vicine.
- **Fase di Reduce**: Ogni reducer riceve dati dai mapper, classificando ogni punto.

Diagramma MapReduce



Vantaggi dell'Approccio Distribuito con Ray

L'implementazione di KNN con MapReduce su Ray offre diversi vantaggi significativi:

- **Scalabilità**: La distribuzione del carico su più nodi permette di scalare l'algoritmo per grandi volumi di dati.
- **Efficienza**: Il parallelismo offerto da Ray accelera il calcolo delle distanze e la determinazione dei vicini più prossimi, riducendo i tempi di esecuzione.

Ambiente di Esecuzione

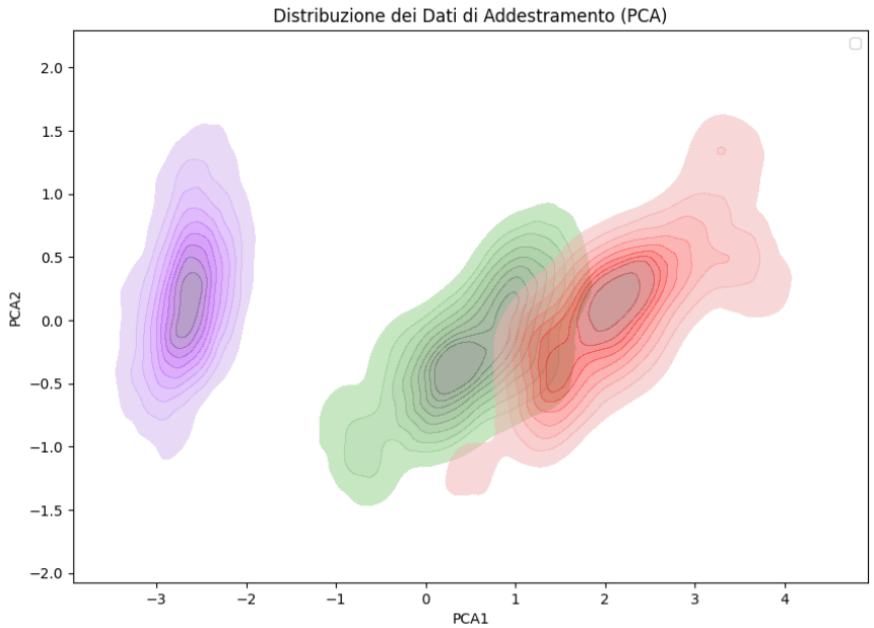
Per l'implementazione del KNN distribuito, è stato utilizzato il cluster del laboratorio, composto da:

- **8 PC**: Configurati per il calcolo distribuito.
- **64 Core in totale**: Per gestire il carico computazionale in parallelo.
- **24 Mapper e 24 Reducer**: Utilizzati nella fase di MapReduce per distribuire e aggregare le operazioni.

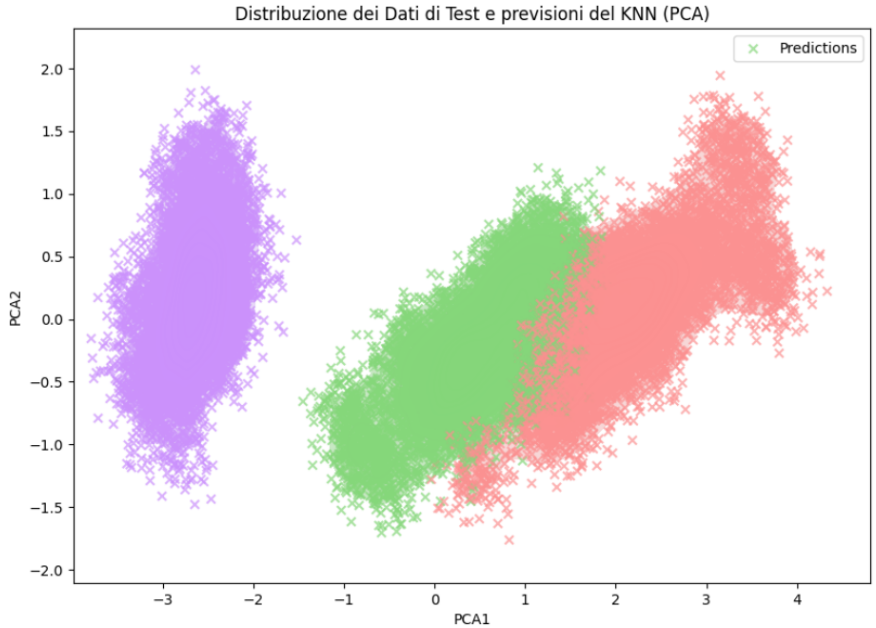
Test effettuato

I dati mostrati sono ottenuti dall'elaborazione del dataset di **90.000 punti** e un valore di $k = 3$, che include dati aumentati e con rumore gaussiano.

Train e previsioni 2D

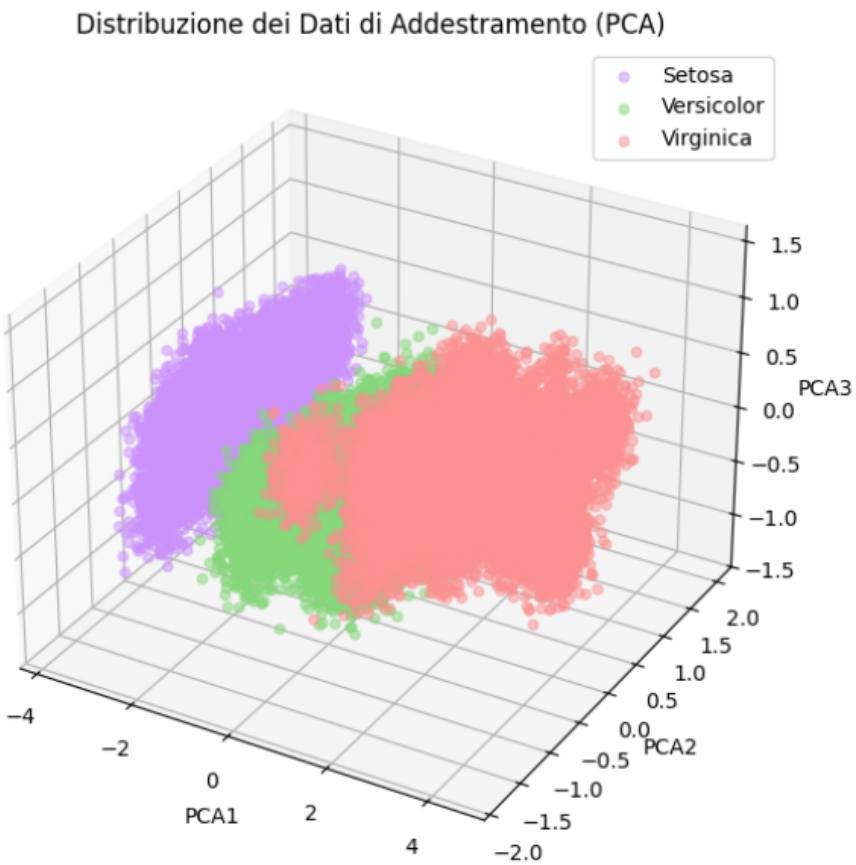


Train set

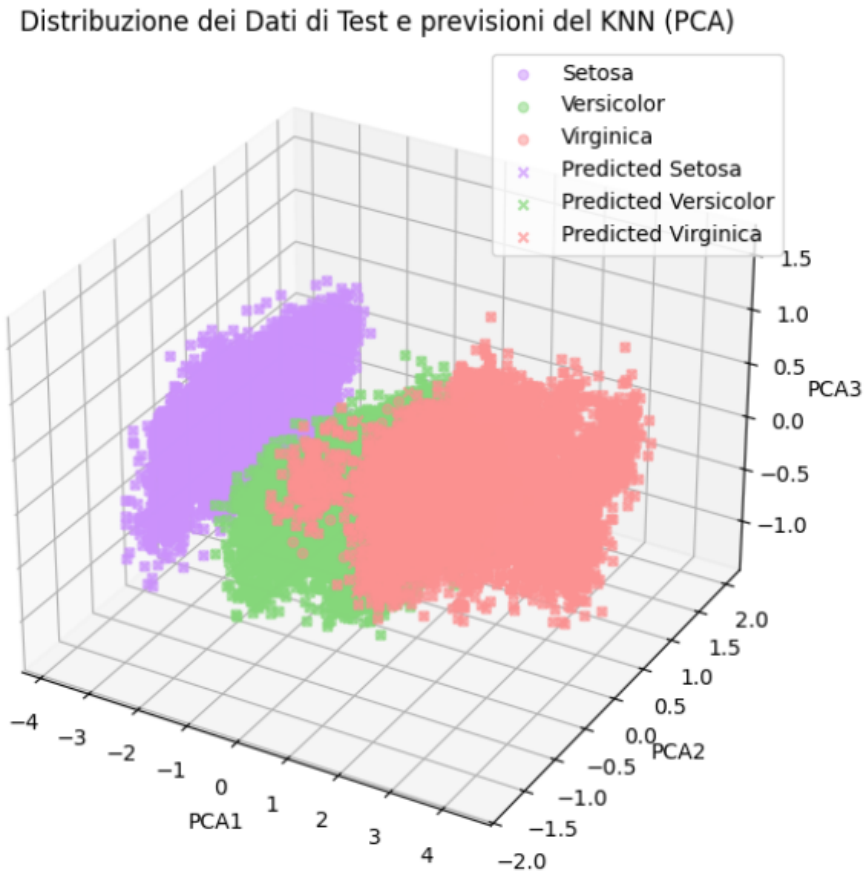


Previsioni del modello

Train e previsioni 3D



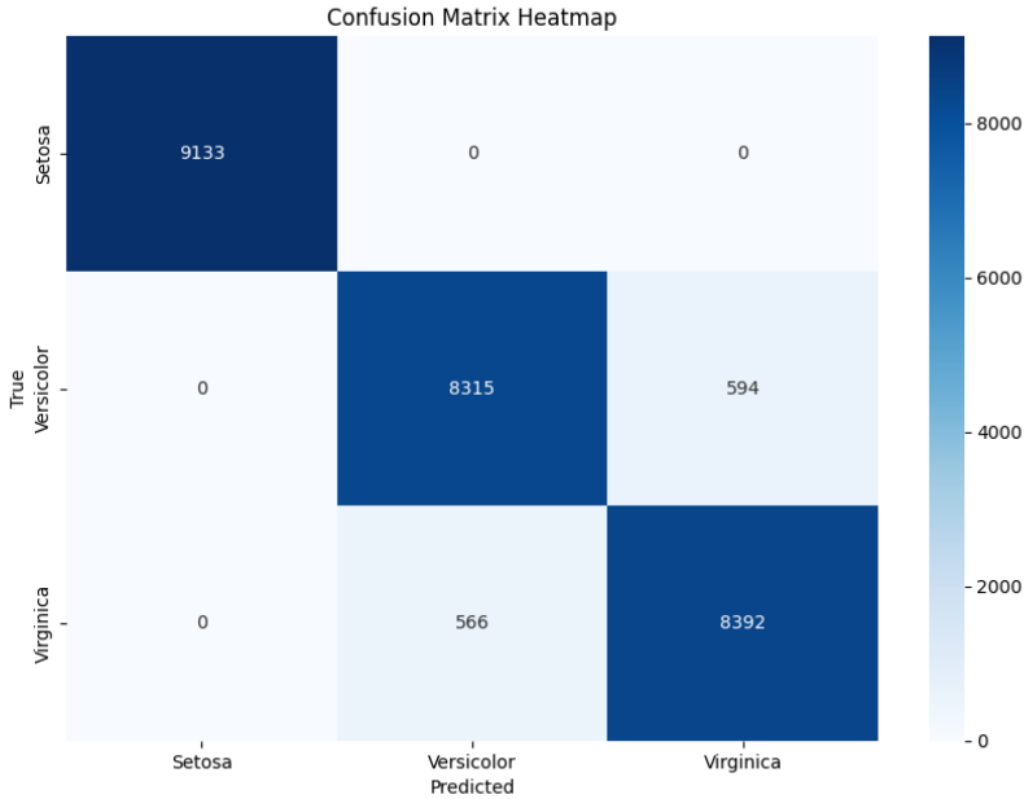
Train set



Previsioni del modello

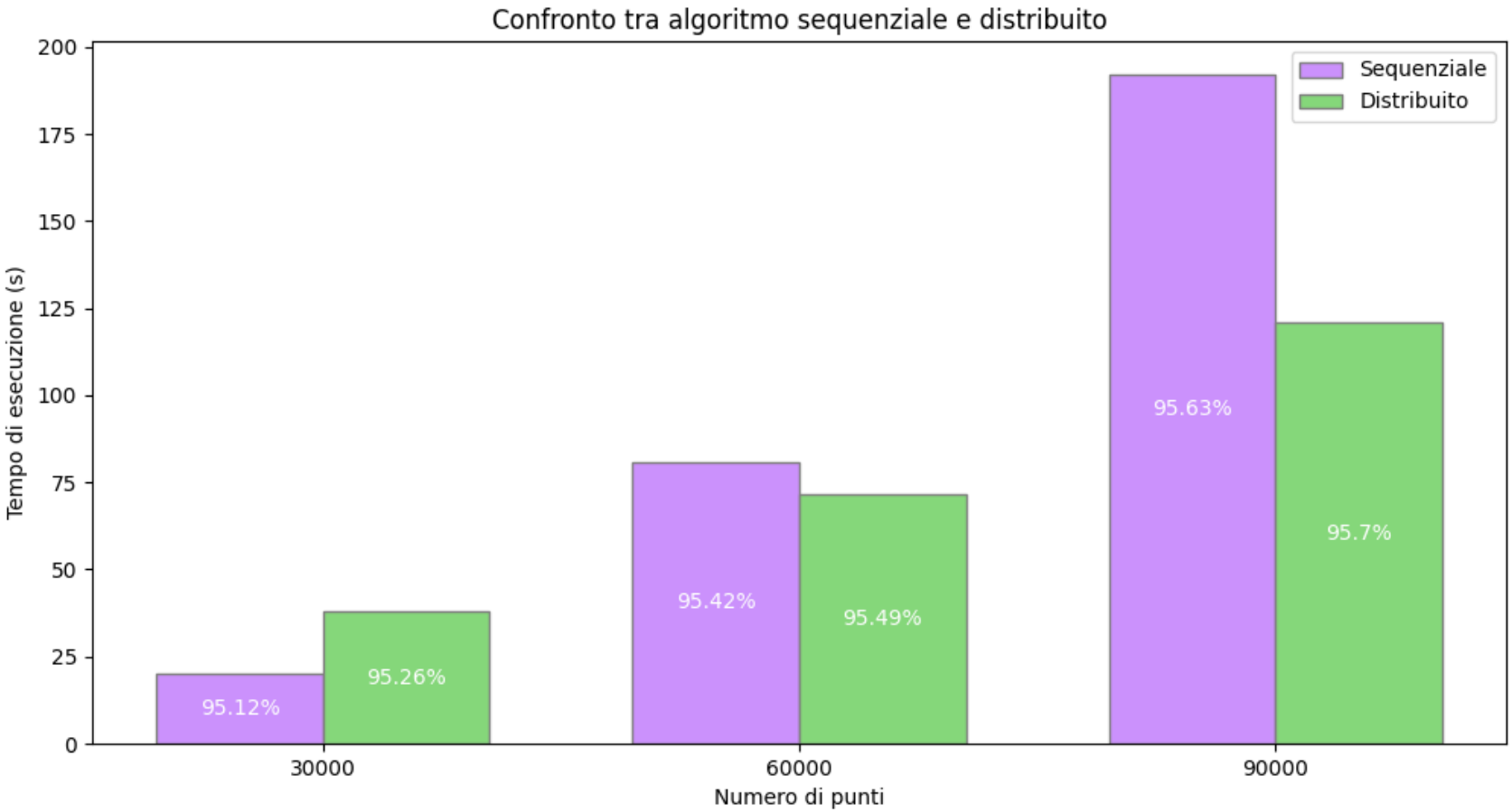
Precisione del modello

Il modello presenta una precisione del 95.70%, si nota inoltre che prevede con precisione massima la classe setosa (cluster viola). Di seguito viene riportata la matrice di confusione.



Confronto algoritmo sequenziale vs distribuito

Per l'esecuzione dell'algoritmo sequenziale è stato utilizzato un PC del laboratorio (8 core 32GB RAM).



Analisi risultati

Dal grafico si evince che:

- L'algoritmo sequenziale è più veloce con pochi punti, questo dimostra i limiti imposti dall'overhead in un approccio distribuito.
- Intorno ai 60.000 punti, l'algoritmo distribuito supera in velocità quello sequenziale (punto di crossover).
- Superato questo punto, l'algoritmo distribuito risulterà sempre più rapido rispetto a quello sequenziale.
- La precisione dell'algoritmo cresce all'aumentare dei dati di train.

Conclusioni

L'approccio distribuito si conferma essere ottimale per un algoritmo computazionalmente intenso come il KNN.
Ricapitolando, il KNN ha dimostrato i suoi pregi e difetti:

- **Pregi riscontrati**: implementazione intuitiva, precisione elevata.
- **Difetti riscontrati**: scelta arbitraria di K, utilizzo eccessivo della memoria.

Sviluppi futuri

L'implementazione può essere sicuramente migliorata con:

- Utilizzo di più macchine.
- Gestione della memoria avanzata.
- Metodo per la risoluzione di un valore K ottimale.

Grazie!