Previsto o non previsto? Previsione delle precipitazioni basate sui dati meteorologici odierni

Template Documentazione Progetto
Terzo appello AA 2022-23.

Gruppo di lavoro

- Giorgia Nardelli, 738746, g.nardelli17@studenti.uniba.it
- Alberttin Mihai Paduraru, 735335, a.paduraru@studenti.uniba.it

URL repo associato: https://github.com/giorgianardelli/ICon-22-23.git

Indice

1. Introduzione

Il problema da noi affrontato consiste nel prevedere se il giorno seguente saranno presenti precipitazioni o meno, avendo a nostra disposizione i dati meteorologici odierni.

Il dataset utilizzato lo si può trovare al seguente link: https://www.kaggle.com/datasets/jsphyg/weather-dataset-rattle-package

Per realizzare tale progetto abbiamo utilizzato algoritmi di machine learning e di apprendimento supervisionato.

La precisione di calcolo avverrà mediante l'uso dei seguenti modelli: regressione logistica, classificatore bayesiano e albero di decisione (mettendoli a confronto).

Le feature di input considerate sono:

- Location (nomi delle Location considerate);
- MinTemp (la Temperatura minima in gradi Celsius);
- MaxTemp (la Temperatura massima in gradi Celsius);
- Rainfall (La quantità di pioggia registrata per il giorno in mm)
- Evaporation (la quantità di acqua trasformata dallo stato liquido in vapore dall'effetto del calore, espressa in mm)
- Sunshine (Il numero di ore di sole splendente durante la giornata)
- WindGustSpeed (La velocità (km/h) della raffica di vento più forte nelle 24 ore fino a mezzanotte)
- WindSpeed9am (Velocità del vento (km/h) in media su 10 minuti prima delle 9:00)
- WindSpeed3pm (Velocità del vento (km/h) in media su 10 minuti prima delle 15:00)
- Humidity9am (Umidità (percentuale) alle 9:00)
- Humidity3pm (Umidità (percentuale) alle 15:00)
- Pressure9am (Pressione atmosferica ridotta al livello medio del mare alle 9 del mattino)
- Pressure3pm (Pressione atmosferica ridotta al livello medio del mare alle 15)
- Cloud9am (Frazione di cielo oscurata da nubi alle 9:00)
- Cloud3pm (Frazione di cielo oscurata da nubi alle 15:00)
- Temp9am (Temperatura in gradi Celsius alle 9:00)
- Temp3pm (Temperatura in gradi Celsius alle 15:00)

- RainToday (Booleano: 1 se la precipitazione (mm) nelle 24 ore fino alle 9 del mattino supera 1 mm, altrimenti 0)

Mentre, verrà utilizzata come feature target RainTomorrow (1 se il giorno seguente piove, 0 altrimenti)

2. Sommario

Inizialmente è stata eseguita una fase di preprocessing sui dati di input che prevede di:

- effettuare un processo di feature selection mediante il metodo 'drop', per le feature non considerate rilevanti.
- Eliminare i valori sconosciuti NaN dalle colonne RainToday e RainTomorrow;
- Riempire i valori NaN delle altre colonne con la media dei valori delle righe che danno lo stesso risultato nella colonna RainTomorrow, mediante il metodo 'mean()'.
- Effettuare l'encoding sulle colonne RainToday, RainTomorrow e Location affinché i modelli possano lavorare nel migliore dei modi;
- Effettuare il bilanciamento dei campioni della feature target in quanto erano molto sbilanciati (come si può vedere nel grafico che abbiamo introdotto per rendere più visibile lo sbilanciamento);
- distinguere le feature di input da quelle di output;
- dividere il dataset in training set e test set.

Successivamente, sono stati utilizzati tre modelli di classificazione differenti basati su apprendimento supervisionato ottenuti dalla libreria Sci-kit learn, utile per effettuare operazioni di machine learning in Python.

I modelli considerati sono:

- Modello Lineare Generalizzato: Regressione Logistica

- Classificatore Probabilistico: Classificatore Bayesiano

- Modello Decisionale: Albero di Decisione

Dopodiché, viene definita l'accuratezza, utilizzando le misure di precision, recall, accuracy ed F1-score sui valori di test e sulla predizione effettuata per ogni modello.

Infine, viene utilizzata come forma di ragionamento la libreria Pytholog per consentire la programmazione logica in Python, creando e popolando una Knowledge Base su cui saranno effettuate delle interrogazioni.

3. Elenco Argomenti di interesse

3.1 Preprocessing

Sommario

In questa sezione viene presentata la fase di preprocessing eseguita sui dati di input per la previsione delle precipitazioni. In particolare, sono stati effettuati diversi passaggi per selezionare le feature rilevanti, gestire i valori mancanti e bilanciare la feature target.

Strumenti utilizzati

Per la fase di preprocessing dei dati sono stati utilizzati diversi strumenti e librerie Python, tra cui Pandas e Scikit-learn.

Decisioni di Progetto

Nella fase di preprocessing, sono state prese diverse decisioni di progetto, tra cui:

- Utilizzo del metodo 'drop' per la feature selection delle variabili non rilevanti
- Rimozione dei valori mancanti NaN dalle colonne RainToday e RainTomorrow
- Encoding delle colonne RainToday, RainTomorrow e Location per migliorare le prestazioni dei modelli di previsione
- Riempimento dei valori mancanti delle altre colonne con la media dei valori delle righe con lo stesso valore nella colonna RainTomorrow, mediante il metodo 'mean()'
- Bilanciamento dei campioni della feature target per gestire lo sbilanciamento dei dati di input
- Divisione del dataset in training set e test set per la valutazione dei modelli

La fase di preprocessing dei dati è stata valutata sulla base di diverse metriche, tra cui l'accuratezza del bilanciamento della feature target e la distribuzione dei dati di input nel training set e nel test set. Inoltre, è stata prodotta una visualizzazione del bilanciamento della feature target per

aiutare la comprensione dei risultati ottenuti, utilizzando la libreria matplotlib.

La fase di bilanciamento dei campioni della feature target, ovvero RainTomorrow. È stata implementata in questo modo:

- viene creato un nuovo DataFrame chiamato df_pioggia che contiene solo le righe del DataFrame originale df in cui la colonna RainTomorrow è uguale a 1, ovvero quando si verifica la pioggia.
- In modo analogo, viene creato un nuovo DataFrame chiamato df_non_pioggia che contiene solo le righe del DataFrame originale in cui la colonna RainTomorrow è uguale a 0, ovvero quando non si verifica la pioggia.
- Successivamente, si bilancia il dataset selezionando un numero di righe del DataFrame df_non_pioggia pari a quello del DataFrame df_pioggia, usando la funzione 'sample' di Pandas. Questo viene fatto per evitare uno sbilanciamento dei campioni durante la fase di addestramento dei modelli di machine learning.
- In particolare, viene campionato casualmente un sottoinsieme delle righe del DataFrame df_non_pioggia con la stessa lunghezza del DataFrame df_pioggia, utilizzando il parametro 'n' della funzione sample.
- L'argomento random_state garantisce che il campionamento sia sempre riproducibile, ovvero che venga ottenuto lo stesso risultato ogni volta che si esegue il codice.
- Andiamo poi a riempire tutti i valori mancanti NaN delle altre feature dei due DataFrame (df_pioggia e df_non_pioggia) mediante il metodo mean(), ovvero con la media.
- Infine, otterremo come risultato un DataFrame dato dalla concatenazione dei due DataFrame (df_pioggia e df_non_pioggia).

L'encoding delle colonne RainToday, RainTomorrow e Location, invece, è stato implementato applicando una mappatura enumerativa univoca per ogni valore distinto di ogni feature.

3.2 Modelli di predizione

SOMMARIO

Sono stati utilizzati tre modelli di predizione: regressione logistica, classificatore Bayesiano e albero di decisione. I dati sono stati pre-

processati e suddivisi in set di training e test. Il modello è stato poi addestrato utilizzando il set di training e utilizzato per fare previsioni sul set di test. Sono state calcolate le metriche di valutazione del modello, inclusa l'accuratezza, la precisione, il richiamo e il punteggio F1.

STRUMENTI UTILIZZATI

Per la regressione logistica e il classificatore Bayesiano sono state utilizzate le implementazioni fornite dalla libreria Python Scikit-learn (in particolare le classi 'LogisticRegression' e 'GaussianNB'), mentre per l'albero di decisione si è utilizzata la classe DecisionTreeClassifier.

In questo caso è stato utilizzato il modello probabilistico Naive Bayes che si basa sull'applicazione del Teorema di Bayes per il calcolo della distribuzione a posteriori $P(X_i|Y)$. Siccome però il calcolo è molto complesso, si applica l'assunzione Naive (ingenua) di indipendenza tra le varie X_i (feature di input) data Y (una certa classe), perdendo quindi la qualità nei risultati, ma rendendo possibile una semplificazione del calcolo.

È stato, inoltre, utilizzato un modello decisionale, ovvero, l'albero di decisione. È un modello di classificazione che usa un albero per rappresentare le scelte che portano a una determinata classe di appartenenza. L'albero viene creato in base alle caratteristiche degli esempi di addestramento, e quindi può essere usato per classificare nuovi esempi in base a tali caratteristiche. L'albero di decisione è una tecnica di classificazione flessibile e interpretabile, utilizzata in molti ambiti come la rilevazione di frodi, la diagnosi medica o la previsione di crediti.

Abbiamo poi utilizzato il modello di regressione logistica, un Modello Lineare Generalizzato (GLM) che viene utilizzato per modellare la probabilità di appartenenza di un'osservazione ad una o più classi, a partire da un insieme di variabili indipendenti. In particolare, la regressione logistica utilizza una funzione logistica per trasformare una combinazione lineare delle variabili indipendenti in un valore compreso tra 0 e 1, che rappresenta la probabilità di appartenenza ad una classe.

Il modello di regressione logistica viene addestrato a partire da un insieme di esempi di addestramento, che consistono in coppie di vettori di feature e di etichette di classe corrispondenti. L'obiettivo dell'addestramento è quello di trovare i coefficienti del modello di regressione logistica che massimizzano la likelihood dei dati di addestramento, ovvero la probabilità di osservare i dati di addestramento dati i parametri del modello.

La regressione logistica è particolarmente utile quando le variabili indipendenti sono continue o discrete, e quando la variabile dipendente è binaria o multinomiale.

DECISIONI DI PROGETTO

Per la regressione logistica e per l'albero di decisione, i parametri predefiniti della libreria sono stati utilizzati senza alcuna modifica. Non sono state prese decisioni specifiche per il classificatore Bayesiano in quanto il modello utilizza solo valori di probabilità.

VALUTAZIONE

Sono state adottate le metriche di accuratezza, precision, recall e F1-score per valutare le prestazioni dei modelli di predizione. Sono stati eseguiti test su un set di dati di prova (20% del dataset originale).

3.3 Accuratezza classificatori

Dopo aver eseguito queste operazioni, è stata calcolata l'accuratezza dei modelli utilizzati rispetto ai valori di test e alla predizione restituita da essi ed è risultato quanto segue:

	Accuracy	Precision	Recall	F1-Score
Regressione Logistica	0.861629677109206	0.8603761555626395	0.8638182109137462	0.8620937475045916
Classificatore Bayesiano	0.8764522073551799	0.8773448773448773	0.8756601056168987	0.8765016818837097
Albero di Decisione	0.832144860187485	0.8463975983989326	0.8121299407905265	0.8289097590853408

3.4 Programmazione Logica

Nel progetto viene utilizzata la libreria 'Pytholog' libreria che consente di lavorare con Prolog in Python.

Mediante essa è stato possibile creare una Knowledge Base chiamata ‘meteo’, contenente informazioni riguardo il meteo nelle diverse città, quali:

- Temperatura (indica la temperatura in gradi in una città);
- km_vento (indica la velocità del vento in km/h in una città);
- umidita_percento (indica la percentuale di umidità di una città);
- condizione_attuale (indica la condizione meteorologica in una città (soleggiato, piovoso, nuvoloso...);
- mm_pioggia (indica la quantità di pioggia in una città espressa in millimetri (mm));
- pioggia_forte (permette di capire se in una città la pioggia è forte o meno, controllando se in quella città piove e se i millimetri di pioggia sono maggiori di 2);
- vento_forte (permette di capire se in una città il vento è forte, lo è se in una città la velocità del vento è maggiore di 40 km/h);
- vento_allerta_gialla (indica se in una città la velocità del vento si trova tra 40 km/h e 60 km/h, in tal caso il vento sarà compatibile con un’allerta gialla);
- vento_allerta_arancione (indica se in una città la velocità del vento si trova tra 60 km/h e 75 km/h, in tal caso il vento sarà compatibile con un’allerta arancione);
- vento_allerta_rossa (indica se in una città la velocità del vento è maggiore di 75 km/h, in tal caso il vento sarà compatibile con un’allerta rossa);
- allerta (indica se in una città è presente o meno un’allerta meteo)
- allerta_gialla (indica se in una città è presente o meno un’allerta meteo gialla);
- allerta_arancione (indica se in una città è presente o meno un’allerta meteo arancione);
- allerta_rossa (indica se in una città è presente o meno un’allerta meteo rossa).

Sulla base di queste sono state effettuate diverse interrogazioni alla KB, da cui si potrà ottenere un valore ‘True’ o ‘False’ nel caso di query decisionali, oppure un insieme di istanze delle variabili che soddisfano la query nel caso di query con variabili.

4. Conclusioni

Le metriche utilizzate per valutare le performance dei modelli sono state accuracy, precision, recall e F1 score. Dopo aver confrontato i diversi modelli, possiamo concludere che, il classificatore Bayesiano ha ottenuto i risultati migliori, con un'accuracy del 87%, circa. Tuttavia, tutti e tre i modelli hanno ottenuto performance simili e sono in grado di effettuare previsioni accurate sulla base dei dati di input forniti.

5. Riferimenti bibliografici

Artificial Intelligence Foundations of Computational Agents - David L. Poole Alan K Mackworth.