UNIVERSITÀ
DEGLI STUDI DI BARI
ALDO MORO

Progetto realizzato da:

- Pietro Walter Soriano (mat. 717033)
- Vincenzo Quagliarella (mat. 745372)

INGEGNERIA DELLA CONOSCENZA:

PROGETTO "SAFE-FISHING"

Obiettivi del progetto:

Il progetto *Safe-Fishing* nasce dall'idea di implementare vari metodi di apprendimento, studiati nel corso di ingegneria della conoscenza, mettendoli a disposizione ai fini di monitorare e regolamentare l'attività di pesca nel Mar Mediterraneo in funzione dell'inquinamento presente.

Avvalendoci della suddivisione del Mar Mediterraneo e Mar Nero, in 10 regioni, regolamentata dalla *Food and Agriculture Organization of the United Nations* (FAO) *Area37*, il software simula il livello di inquinamento di ogni zona, su una scala di cinque valori, che viene determinato sulla base del valore dei fattori inquinanti.
Il sistema si occupa di osservare e fornire all'utente, tramite predizioni, le probabilità di pesca, di alcune razze di pesci, nelle varie zone, in questo modo l'utente è in grado di capire qual è la miglior zona per effettuare attività di pesca per quel determinato pesce, ed inoltre gli presenterà il percorso ideale, qualora ci fosse, per raggiungere una determinata zona, evitando zone altamente inquinate con lo scopo di non peggiorare l'inquinamento di tali zone ed evitare di pescare pesce inquinato nella zona FAO in cui si è diretti.
Questo meccanismo, quindi, fa in modo che l'utente dia priorità alle zone meno inquinate e che le preferisca alle zone migliori per l'attività pesca ma molto inquinate; infatti, l'algoritmo di ricerca si occupa di trovare il percorso meno inquinato da percorrere, qualora ci fosse un percorso effettivamente non molto inquinato. L'utente attraverso le funzionalità presenti del sistema sarà in grado di trovare percorsi ideali, verificare se può passare da una zona all'altra, verificare il livello di inquinamento di ogni zona. Il progetto è stato realizzato per far fronte ad alta dinamicità dei dati; infatti, i valori di inquinamento sono assegnati in modo randomico in modo tale da evidenziare che potrebbero tranquillamente essere, in un applicativo reale, molto variabili ma questo non potrà ridurre in alcun modo le capacità del software e la sua affidabilità. In un'applicazione reale la variabilità dei valori di inquinamento varia in modo decisamente più lento all'assegnazione random di valori ad ogni esecuzione del software.

Argomenti sviluppati:

- Base di conoscenza con interrogazioni;
- Classificatore con albero decisionale e relativa valutazione;
- Grafo pesato con algoritmo di ricerca “A*”

Strumenti di sviluppo:

Il progetto è stato interamente sviluppato in *Python* in quanto linguaggio di programmazione semplice, veloce e flessibile. Abbiamo aggiunto le seguenti librerie Python per poter effettuare con maggior semplicità alcune operazioni utili per i nostri obiettivi:

- *Pillow*. Libreria utilizzata per il processing di immagini.
- *Pandas*. Libreria utilizzata per la manipolazione ed analisi di dati.
- *Scikit-learn*. Libreria utilizzata per implementare diversi algoritmi di apprendimento classificatori o regressori. Contiene diversi dataset didattici di addestramento non utilizzati per questo progetto.
- *Numpy:* Libreria che dispone di una vasta quantità di funzioni matematiche, tra cui la funzione random utilizzata per il nostro progetto.

Base di conoscenza con interrogazioni:

La *knowledge-base* è stata implementata con la logica del primo ordine. Una base di conoscenza in logica di primo ordine è costituita da un insieme di proposizioni, dette assiomi, che sono assunte essere vere senza dimostrazione. La base di conoscenza è utile per rappresentare, all’interno di una macchina, la conoscenza riguardo un particolare mondo.

L'utente potrà chiedere se una zona in particolare avrà un determinato valore di inquinamento, chiedendo al software di illustrare il procedimento tramite comando «how»; oppure l'utente potrà chiedere se si può navigare da una zona ad un'altra, con la medesima possibilità di chiedere al software la spiegazione della scelta digitando il comando opportuno.

Classificatore e valutazione:

Il classificatore scelto è il *decison-tree* o albero decisionale, il quale è stato implementato in modo tale che all'utente, durante l'esecuzione del programma, venga chiesto in input la zona dove voler effettuare l'attività di pesca e la razza di pesce che si è interessati a pescare, il sistema sulla base di queste scelte fornirà in output la probabilità specifica per quella razza di pesce in quella zona.

Il sistema farà quindi predizioni grazie al fatto che le probabilità di pesca decrescono man mano che ci si muove nelle zone più ad est.
Ad esempio se volessi pescare un tonno, sarebbe molto più probabile pescarlo nella zona 1.1 che pescarlo nella zona 4.3, resta il fatto che delle zone possono essere inaccessibili a causa dell'inquinamento molto alto anche qualora avessero una probabilità di pesca molto alta.
L'utente, in base alla probabilità, deciderà quanto prolifica può essere l'attività di pesca nella zona, dovrà usufruire però delle altre funzionalità disponibili dal sistema per capire se tale zona è accessibile e per verificare il suo livello di inquinamento, ma non dovrà farlo obbligatoriamente. La valutazione viene effettuata attraverso il *k-fold cross validation* in quanto il dataset utilizzato raggiunge un valore minimo di diverse tuple adeguato a permettere un efficiente funzionamento.
Il numero di fold scelto è pari a 5, valore standard della libreria che lo gestisce.
Il cross val score a sua volta utilizza come parametro di scoring l'R^2 ovvero il coefficiente di determinazione, un indice che misura il legame tra la variabilità dei dati e la correttezza del modello utilizzato.

Grafo pesato con algoritmo di ricerca "A*":

Il grafo implementato ha lo scopo di individuare un percorso da un dato nodo iniziale verso un nodo *goal.* Per effettuare tale ricerca è stato scelto di implementare l'algoritmo di ricerca chiamato "A*". Nel grafo in questione i nodi sono costituiti dalle regioni FAO e gli archi sono i collegamenti diretti con le zone confinanti.
L' utente ha la possibilità di ricercare un percorso che non sia molto inquinato in modo tale da permettergli di non aumentare i valori di inquinamento delle zone già sufficientemente inquinate ed evitare di pescare pesce inquinato qualora ci fosse il bisogno di effettuare attività di pesca anche in regioni di transito. Gli archi, quindi, sono pesati ed il relativo peso è il valore dei fattori inquinanti, determinanti per l'assegnamento del livello di inquinamento. Ogni arco collega regioni confinanti. I livelli di inquinamento vengono raccolti in una scala di 5 valori: molto basso, basso, moderato, alto, molto alto.
In base a tali livelli, il sistema valuterà la possibilità o meno di spostarsi da una zona FAO di pesca ad un'altra. Non è possibile muoversi da e verso le regioni che hanno un valore di inquinamento "molto alto", in quanto l'intento è di effettuare una pesca sicura, lontana dal rischio di pescare pesce altamente inquinato. In caso il valore di inquinamento sarà diverso da "molto alto" allora ci si potrà spostare o attraversare la zona FAO interessata.

L'algoritmo di ricerca si occuperà di trovare il percorso meno costoso (e quindi meno inquinato) da percorrere.

Funzionamento comandi:

- *classificazione*. Esegue il classificatore;
- *valutazioneClassificatore*. Esegue valutazione classificatore;
- *mostraZoneFAO*. Mostra immagine con zone FAO distribuite sulla mappa
- *trovaZoneInquinamentoMoltoBasso*. Trova le zone FAO con inquinamento “molto basso”
- *trovaZoneInquinamentoBasso*.
- *trovaZoneInquinamentoModerato*.
- *trovaZoneInquinamentoAlto*.
- *trovaZoneInquinamentoMoltoAlto*.
- *visualizzaListaInquinamenti*. Stampa la lista dei possibili valori di inquinamento che le zone FAO possono avere.
- *trovaPercorso*. Trova un percorso ottimale, qualora esistesse, tra zona Start e zona Goal.
- *verificaPassaggio*. Verifica il passaggio da una zona Start ad una zona Goal.
- *verificaInquinamentoZona*. Verifica se una determinata zona ha un determinato valore di inquinamento.
- *esci*. Chiude il programma.

Link repository:

https://github.com/PWS00/Progetto-iCon---finale