Detectarea anomaliilor folosind Machine Learning

Tehnici de învățare automata

Proiect realizat de Mitria Alexandru si Pirvan Teodora-Maria

Grupa 334AC si 333AC

Facultatea de Automatică și Calculatoare

Universitatea Politehnică din București

Contents

1. Introducere 3
   1.1. Context si motivatie 3
   1.2. Obiectivele si scopul proiectului 3
2. Fundament si Context Teoretic 4
   2.1. Descrierea problemei si a modului de lucru 4
   2.2. Motivatia alegerii algoritmului Support Vector Machine 5
3. Prezentarea si descrierea algoritmului 5
   3.1. Informatii furnizate 5
   3.2. Parcurgerea si explicarea codului 6
4. Rezultate 8
5. O scurta comparatie 9

1. Introducere

1.1. Context si motivatie

Contextul general al proiectului se situează în domeniul medical, unde ne confruntăm cu o problemă semnificativă privind detectarea precoce a anomaliilor corpului uman. Pentru o analiza exacta si eficienta, o multitudine de date este supusa unor serii de testari ce pot prezenta diverse intarzieri sau erori.

1.2. Obiectivele si scopul proiectului

Scopul acestui proiect este de a analiza un set complex de date atribuite unui numar finit de pacienti, realizandu-se ulterior o clasificare precisa a anomaliilor inregistrate. Astfel, fiind luate in considerare sase caracteristici ale pacientilor (incidenta, inclinarea, unghiul, panta, raza și gradul), se vor clasifica mostrele ca fiind normale sau anormale. De asemenea, probele anormale corespund pacienților cu diferite probleme medicale (de exemplu, hernie de disc, spondilolisteză). Programul utilizeaza metode de analiza ce fac parte din Machine Learning, algoritmul utilizat fiind Support Vector Machine, SVM.

Cu ajutorul acestui proiect, se propune atingerea urmatoarelor obiective:

- Facilitate in rezolvarea problemelor medicale – Analiza unui set de date amplu, respectiv o clasificare succinta in functie de acestea, se poate demonstra a fi destul de dificila in lipsa unui program antrenat pentru asemenea situatii. In acest sens, implementarea unui model ce se concentreaza asupra prelucrarii unor astfel de informatii eficientizeaza intregul proces, exemplificand o atentie meticuloasa la detalii.

- Optimizarea eficientei – Modelul propus in scanarea caracteristicilor pacientilor, respectiv clasificarea exacta a problemelor (anomaliilor) acestora, se poate dovedi foarte eficient in ceea ce priveste economisirea de resurse. Asigurarea rapiditatii si eficientei algoritmului, în special în gestionarea unor seturi mari de date mari, contribuie la economisirea timpului si a mijloacelor oferite.

- Acuratetea informatiilor returnate – Cu ajutorul unui algoritm puternic antrenat si testat, se poate confirma acuratetea si fiabilitatea programului in identificarea anomaliilor.Astfel, este imposibila eroarea diagnosticului si existenta incertitudinilor privind analiza datelor.

- Evaluarea performantelor – Programul este indeajuns de antrenat astfel incat sa evalueze performantele pe un anumit set de date speciale. I se verifica, astfel, acuratetea generala in ceea ce priveste modul de detectare a anomaliilor, clasificarea corecta a lor (situatii obisnuite/ neobisnuite) si este asigurata functionarea sa optima.

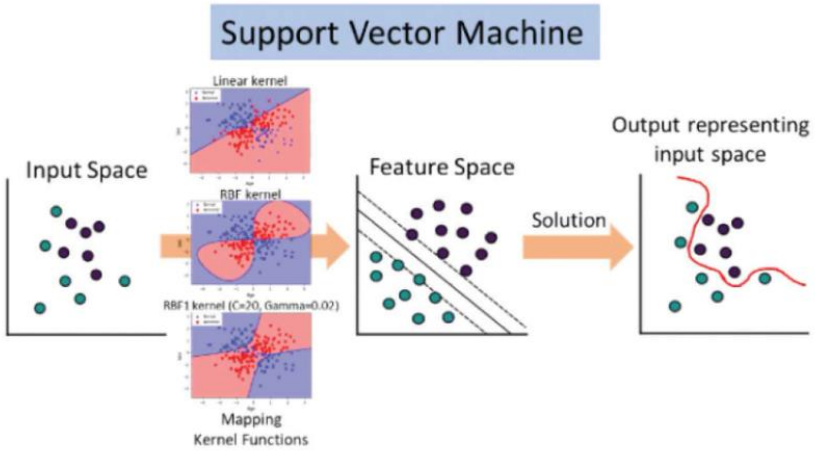
Asadar, programul utilizeaza tehnici de invatare automata bazate pe conceptul de "Machine Learning", avand la baza un algoritm eficient in ceea ce priveste prelucrarea si clasificarea unui set de date amplu.

2. Fundament si Context Teoretic

2.1. Descrierea problemei si a modului de lucru

Problematica se concentreaza in jurul actualizarii si facilitarii diferitelor servicii din interiorul domeniului medical, fapt ce necesita implicarea din punct de vedere ingineresc. In acest caz, programul propus lucreaza in detectarea anomaliilor dintr-un set complex de date, algoritmul de baza fiind Support Vector Machine.

Algoritmul SVM a fost considerat potrivit din punct de vedere al mecanismului de procesare al informatiilor si al rezultatelor propuse ulterior. Acesta este un algoritm robust de învățare automată supravegheat, utilizat în principal pentru sarcini de clasificare binară, în care scopul este de a clasifica punctele de date într-una din două clase, de obicei etichetate ca +1 și -1. SVM își propune să găsească un hiperplan care separă cel mai bine datele în aceste două clase, maximizând în același timp marja dintre ele.



2.2. Motivatia alegerii algoritmului Support Vector Machine

- Capacitate de gestionare a seturilor ample de date – Acest algoritm functioneaza eficient in lucrul cu seturi de date cu mulți vectori de caracteristici sau cu dimensiuni ridicate ale caracteristicilor, asemanatoare acestui caz.

- Flexibilitate în utilizarea kernel-urilor: SVM oferă opțiuni pentru a folosi diverse kernel-uri, cum ar fi kernelul liniar, RBF (Radial Basis Function), polinomial etc. Acest lucru îi permite să trateze seturi de date cu separări non-liniare.

- Bune rezultate pentru clasificarea binară: SVM este frecvent utilizat pentru sarcini de clasificare binară, cum ar fi detectarea spam-ului, clasificarea imaginilor sau detectarea de anomalii.
- Eficiență în spațiul de caracteristici de înaltă dimensiune: SVM utilizează o tehnică numită "trucul kernel" pentru a calcula produsele scalare într-un spațiu de caracteristici de înaltă dimensiune fără a efectua explicit transformări costisitoare ale datelor.

- Configurare usoara – Implementarea algoritmului este foarte usoara, aceasta realizandu-se prin descarcarea librariei “sklearn.svm”, iar ulterior importarea functiei care este deja configurata(“SVC”). De asemenea, in cazul aparitiei vreunei probleme sau neconcordante in interiorul programului, acesta poate fi foarte usor modificat.

- Rezistenta la overfittind – Acest algoritm se prezinta mult mai rezistent la overfitting decat altele, avand o performanta generala foarte buna.

3. Prezentarea si descrierea algoritmului

3.1. Informatii furnizate

Au fost oferite doua fisiere (test, respectiv train) cu extensia “.csv”, ce include in componenta lor coloane ce reprezinta urmatoarele:

- Id (tip int) – lucreaza sub forma unei chei primare de identificare al fiecarui set de date (pacient)
- 6 feature-uri (tip float) – caracteristici ce indica valoarea diferitelor elemente precum: incidenta, inclinarea, unghiul, panta, raza și gradul
- Is_anomaly (binar: 0 sau 1) – varianta ce indica existenta anomaliei: 0 daca nu exista, 1 daca exista. De asemenea, aceasta va fi incarcata in fisierul final in cadrul fiecarui Id.

Informatiile finale (Id, is_anomaly) sunt stocate intr-un fisier tot de tip .csv numit submission/ submission_updated, ce va fi ulterior incarcat pe platforma Kaggle pentru clasificare, respectiv punctare. De asemenea, acuratetea si eficacitatea proiectului a fost evaluata cu ajutorul platformei respective.

```
10  import pandas as pd
11  from sklearn.preprocessing import StandardScaler
12  from sklearn.svm import SVC
```

Biblioteci utilizate:

- “pandas” sub aliasul “pd” este o biblioteca necesara in gestionarea unui set de date amplu, cu ajutorul acesteia fiind realizata citirea fisierelor de date cu extensia .csv
- “StandardScaler” apartine bibliotecii “sklearn.preprocessing” si se ocupa cu stangardizarea datelor
- Biblioteca “sklearn.svm” a fost importata datorita necesitatii existentei clasei SVC. Cu ajutorul acesteia se crea modelul de clasificare SVM cu kernel RBF

3.2. Parcurgerea si explicarea codului

Limbajul de programare in care a fost realizata aplicatia este Python3.

```
14  # Încărcarea datelor de antrenament
15  train_data = pd.read_csv('/content/train.csv')
16
17  # Separarea caracteristicilor și a etichetei țintă
18  X_train = train_data.drop(['id', 'is_anomaly'], axis=1)
19  y_train = train_data['is_anomaly']
```

Un prim pas in realizarea cu succes a analizei informatiilor il reprezinta citirea fisierului ce contine datele. Astfel, informatiile din fisierul “train.csv” sunt incarcate in dataframe-ul (o structudra de date) “train_data” .

Ulterior, in “X_train” sunt eliminate coloanele “id” si “is_anomaly”, iar din “y_train” este extrasa eticheta tinta “is_anomaly”.

```
21  # Standardizarea caracteristicilor
22  scaler = StandardScaler()
23  X_train = scaler.fit_transform(X_train)
```

Functia StandardScaler initializeaza un obiect, ce va fi apoi util in standardizarea datelor de antrenament.

```
25  # Inițializarea modelului SVM
26  model = SVC(kernel='rbf', class_weight='balanced', probability=True)
27  model.fit(X_train, y_train) # Antrenarea modelului SVM pe datele de a
```

Se initializeaza modelul SVM cu ajutorul clasei SVC in interiorul careia sunt indicati urmatorii parametri:

- kernel = ‘rbf’ (kernelul radial = datele nu sunt liniar separabile) – este specificat tipul kernelului folosit in interiorul algoritmului SVM.
- class_weight = 'balanced’ – se acorda atentie asupra gestionarii dezechilibrului dintre clase in setul de date.
- probability = True – in locul claselor de predictii, pot fi furnizare estimari de probabilitate.

Acesta este apoi antrenat pe datele de antrenament propuse.

```
29  # Încărcarea datelor de test
30  test_data = pd.read_csv('/content/test.csv')
31
32  # Separarea caracteristicilor pentru datele de test
33  X_test = test_data.drop(['id'], axis=1)
34  X_test = scaler.transform(X_test)
```

Asemanator procedurii initiale de gestionare a datelor de antrenament, sunt incarcate datele de test si se realizeaza mecanisme identice, insa din “X_test” este eliminata numai coloana ‘id’.\

```
36  # Realizarea predicțiilor pe datele de test
37  test_predictions = model.predict(X_test)
```

Asupra modelului SVM (creat cu ajutorul clasei SVC, avand kernel RBF) se realizeaza predictiile pe datele de test.

```
39  # Crearea unui DataFrame pentru submisie
40  submission = pd.DataFrame({ 'id': test_data['id'], 'is_anomaly': test_predictions})
```

Este creat un Dataframe in care sunt stocate id-urile pacientilor, respectiv rezultatele analizate de program salvate in “test_predictions”.

```
45  # Salvarea DataFrame-ului de submisie într-un fișier CSV
46  submission.to_csv('submission.csv', index=False)
```

Dataframe-ul cu rezultate finale (id-urile in concordanta cu rezultatul adiacent acestuia) este incarcat in fisierul .csv, in interiorul coloanei ‘is_anomaly’ existand doar “raspunsuri” binare – 0 daca nu este anomalie, 1 daca este anomalie.

4. Rezultate

Rezultatele finale sunt vizibile in fisierul “submission.csv”, unse de poate observa o clasificare succinta a informatiilor oferite.

	A	B	C	D	E
1	id	is_anomaly			
2	76	0			
3	15	1			
4	196	0			
5	77	0			
6	166	0			
7	222	1			
8	224	1			
9	154	0			
10	226	1			
11	99	0			
12	187	0			
13	19	0			
14	0	0			
15	13	0			
16	186	0			
17	68	1			
18	25	0			
19	130	0			
20	201	0			
21	40	0			
22	64	0			
23	197	0			
24	147	0			
25	152	0			
26	103	0			
27	211	0			
28	14	0			
29	151	0			
30	155	0			

5. O scurta comparatie

Putem compara algoritmul folosit SVM cu algoritmul IsolationForest in ceea ce priveste eficienta si gestionarea seturilor de date propuse in cadrul acestui proiect de laborator.

Este cunoscut faptul ca IsolationForest este un bun program in ceea ce priveste administrarea informatiilor ample (asemenea lui SVM) in detectarea anomaliilor. Cu ajutorul acestuia a fost realizata o prima abordare a proiectului, insa a fost cat se poate de vizibil ca randamentele celor doua sunt foarte diferite. In timp ce SVM propunea pe platforma Kaggle un scor de aproximativ 0.85, IsolationForest nu putea trece peste pragul de 0.65 (cel putin, nu in urma multiplelor incercari ale noastre).

Asadar, pentru acest caz, algoritmul SVM a fost considerat o alegere mult mai optima din punct de vedere a rezultatelor.