Documentație proiect IA

Totolici Alexandru-Gabriel, grupa 243

Clasificatorul Naive-Bayes

Acest clasificator calculează probabilitățile de apartenență la fiecare clasă pentru un set de date de intrare, folosind teorema Bayes și presupunerea de independență între caracteristici.

Date

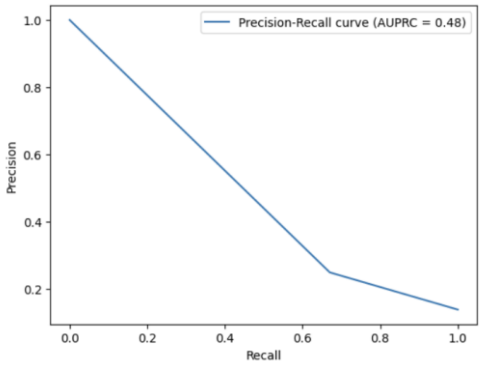
Pentru a normaliza datele, am folosit librăriile cv2 și skimage, astfel testând pe mai multe variante ale imaginilor: mai multe formate de dimensiune (100x100, 125x125, 150x150), cu 3 canale (RGB) și cu 1 singur canal (grayscale), sau valorile pixelilor normalizate între 0 și 1, dar cea mai bună variantă care a funcționat pentru modelul meu a fost ca imaginea să fie pe dimensiunea 100 x 100 cu un singur canal.

100 x 100 px

```
F1 Score:
 0.3627450980392157
Confusion Matrix:
 [[1165  559]
 [  91  185]]
Classification Report:
              precision    recall  f1-score   support

           0       0.93      0.68      0.78      1724
           1       0.25      0.67      0.36       276

    accuracy                           0.68      2000
   macro avg       0.59      0.67      0.57      2000
weighted avg       0.83      0.68      0.72      2000
```

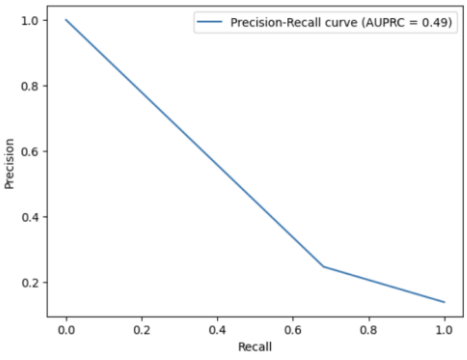


150 x 150 px

```
F1 Score:
 0.36153846153846153
Confusion Matrix:
 [[1148  576]
 [  88  188]]
Classification Report:
              precision    recall  f1-score   support

           0       0.93      0.67      0.78      1724
           1       0.25      0.68      0.36       276

    accuracy                           0.67      2000
   macro avg       0.59      0.67      0.57      2000
weighted avg       0.83      0.67      0.72      2000
```



Rezultate

În clasificarea Bayes Naive, parametrul alpha este un hiperparametru utilizat pentru a controla aproximarea probabilităților de apartenență unei imagini la o clasă.

Pentru acest model, balansarea setului de date prin undersampling nu a avut niciun efect semnificativ asupra performanței modelului.

În urma testelor, am observat că valoarea potrivită a hiperparametrului alpha pentru modelul meu este alpha = 0.1. Crescând valoarea parametrului alpha, f1_score scade în felul următor:

- alpha = 0.1 score = 0.366

```
F1 Score:
 0.3666026871401152
Confusion Matrix:
 [[1149  575]
 [  85  191]]
Classification Report:
              precision    recall  f1-score   support

           0       0.93      0.67      0.78      1724
           1       0.25      0.69      0.37       276

    accuracy                           0.67      2000
   macro avg       0.59      0.68      0.57      2000
weighted avg       0.84      0.67      0.72      2000
```

- alpha = 0.5 score = 0.363

```
F1 Score:
 0.36346153846153845
Confusion Matrix:
 [[1149  575]
 [  87  189]]
Classification Report:
              precision    recall  f1-score   support

           0       0.93      0.67      0.78      1724
           1       0.25      0.68      0.36       276

    accuracy                           0.67      2000
   macro avg       0.59      0.68      0.57      2000
weighted avg       0.84      0.67      0.72      2000
```

- alpha = 10 f1_score = 0.342

```
F1 Score:
 0.34270650263620384
Confusion Matrix:
 [[1057  667]
 [  81  195]]
Classification Report:
              precision    recall  f1-score   support

           0       0.93      0.61      0.74      1724
           1       0.23      0.71      0.34       276

    accuracy                           0.63      2000
   macro avg       0.58      0.66      0.54      2000
weighted avg       0.83      0.63      0.68      2000
```

- alpha = 100 f1_score = 0.3

```
F1 Score:
 0.3062015503875969
Confusion Matrix:
 [[ 689 1035]
 [  39  237]]
Classification Report:
              precision    recall  f1-score   support

           0       0.95      0.40      0.56      1724
           1       0.19      0.86      0.31       276

    accuracy                           0.46      2000
   macro avg       0.57      0.63      0.43      2000
weighted avg       0.84      0.46      0.53      2000
```

În concluzie, cu clasificatorul Naive-Bayes am putut ajunge la f1_score maxim de aproximativ 0.36, după ce am normalizat datele de antrenare și testare și am încercat diferite combinații pentru hiperparametrii modelului testat. Înainte de a efectua aceste modificări, f1_score maxim era de aproximativ 0.3.

Random Forest Classifier

RFC este un algoritm de învățare supervizată, construit pe baza mai multor arbori de decizie.Fiecare arbore de decizie este creat prin extragerea aleatorie a unui set de caracteristici din setul de date și alegerea caracteristicii optime pentru a diviza setul de date în mai multe subseturi.Folosind mai mulți arbori de decizie, modelul combină predicțiile fiecărui arbore pentru a obține o predicție finală.

Date

Pentru a normaliza datele, am folosit librăriile cv2 și skimage, astfel testând pe mai multe variante ale imaginilor: mai multe formate de dimensiune (100x100, 125x125, 150x150), cu 3 canale (RGB) și cu 1 singur canal (grayscale), sau valorile pixelilor normalizate între 0 și 1, dar cea mai bună variantă care a funcționat pentru modelul meu a fost ca imaginea să fie pe dimensiunea 125 x 125 cu un singur canal.

Rezultate

Pentru început, am implementat un model standard care clasifică imaginile de test în cele 2 categorii, iar f1_score maxim pe care am putut să-l obțin fără preprocesarea datelor și fără a îmbunătăți parametrii modelului a fost ~0.44.

outputv3.csv Complete · 14d ago	0.48879	0.42201
outputv2.csv Complete · 14d ago	0.46718	0.43192
output.csv Complete · 14d ago	0.5037	0.44545

Următorul pas a fost echilibrarea setului de date. Undersampling-ul este procedeul prin care reducem numărul de elemente din clasa majoritară, astfel încât numărul ambelor clase să fie aproximativ egal. În cazul de față, setul de date de antrenare era compus din 85% imagini din clasa 0 și 15% imagini din clasa 1. Astfel, am restrâns setul de date, păstrând toate elementele din clasa 1, și doar 2000 de elemente din clasa 0.

Astfel, acuratețea modelului meu a crescut, precum și f1_score până aproape de 0.48.

```
Accuracy on validation set: 0.876
Precision on validation set: 0.5729166666666666
Recall on validation set: 0.39855072463768115
F1 Score:
 0.47008547008547
Confusion Matrix:
 [[1642   82]
 [ 166  110]]
Classification Report:
              precision    recall  f1-score   support

           0       0.93      0.67      0.78      1724
           1       0.25      0.68      0.36       276

    accuracy                           0.67      2000
   macro avg       0.59      0.68      0.57      2000
weighted avg       0.84      0.67      0.72      2000
```

Următorul pas a fost să testez mai multe valori ale parametrilor modelului prezentat.

max_depth = Adâncimea maximă a unui arbore.

n_estimators = Numărul de arbori dintr-o pădure.

max_features = Numărul maxim de caracteristici (features) care sunt luate în considerare în fiecare nod atunci când se construiește un arbore.

1. n_estimators=100, max_depth=10, max_features='sqrt'
2. n_estimators=50, max_depth=5, max_features='auto'
3. n_estimators=200, max_depth=15, max_features=0.5
4. n_estimators=50, max_depth=10, max_features=0.2
5. n_estimators=300, max_depth=18, max_features=0.8

max_depth	n_estimators	max_features	F1 Score
5	50	auto	0.562
10	50	0.2	0.524
10	100	sqrt	0.527
15	200	0.5	0.512
20	100	log2	0.529

max_depth=5, n=50, max_features = auto || max_depth=10, n=50, max_features = 0.2

```
F1 Score:
 0.5627705627705628
Confusion Matrix:
 [[1502  222]
 [  81  195]]
Classification Report:
              precision    recall  f1-score   support

           0       0.95      0.87      0.91      1724
           1       0.47      0.71      0.56       276

    accuracy                           0.85      2000
   macro avg       0.71      0.79      0.74      2000
weighted avg       0.88      0.85      0.86      2000

Accuracy : 0.8485
Precision : 0.4676258992805755
Recall : 0.7065217391304348
```

```
F1 Score:
 0.5247148288973384
Confusion Matrix:
 [[1418  306]
 [  69  207]]
Classification Report:
              precision    recall  f1-score   support

           0       0.95      0.82      0.88      1724
           1       0.40      0.75      0.52       276

    accuracy                           0.81      2000
   macro avg       0.68      0.79      0.70      2000
weighted avg       0.88      0.81      0.83      2000

Accuracy : 0.8125
Precision : 0.40350877192982454
Recall : 0.75
```

max_depth=10, n=100, max_features = sqrt || max_depth=20, n=100, max_features = log2

```
F1 Score:
 0.5272727272727273
Confusion Matrix:
 [[1433  291]
 [  73  203]]
Classification Report:
              precision    recall  f1-score   support

           0       0.95      0.83      0.89      1724
           1       0.41      0.74      0.53       276

    accuracy                           0.82      2000
   macro avg       0.68      0.78      0.71      2000
weighted avg       0.88      0.82      0.84      2000

Accuracy : 0.818
Precision : 0.4109311740890688
Recall : 0.7355072463768116
```

```
F1 Score:
 0.5297691373025516
Confusion Matrix:
 [[1395  329]
 [  58  218]]
Classification Report:
              precision    recall  f1-score   support

           0       0.96      0.81      0.88      1724
           1       0.40      0.79      0.53       276

    accuracy                           0.81      2000
   macro avg       0.68      0.80      0.70      2000
weighted avg       0.88      0.81      0.83      2000

Accuracy : 0.8065
Precision : 0.39853747714808047
Recall : 0.7898550724637681
```

În urma mai multor teste, am observat că adăugând mai mulți arbori modelului meu prin parametrul n_estimators, timpul de execuție crește considerabil, iar acuratețea și f1_score scad când n_estimators ia valori mai mari decât 50. Astfel, cel mai mare f1_score obținut a fost cu modelul următor:

```
rfc = RandomForestClassifier(n_estimators=50,
                             max_depth=5,
                             max_features='sqrt',
                             class_weight='balanced')
```

```
Accuracy on validation set: 0.8575
Precision on validation set: 0.488
Recall on validation set: 0.6630434782608695
F1 Score:
 0.5622119815668203
Confusion Matrix:
 [[1532  192]
 [  93  183]]
Classification Report:
```

În concluzie, cu un model de tip RFC am putut ajunge la f1_score maxim de aproximativ 0.56, după ce am normalizat datele de antrenare și testare, am balansat setul de date prin undesampling și am încercat diferite combinații pentru hiperparametrii modelului testat. Înainte de a efectua aceste modificări, f1_score maxim era de aproximativ 0.4-0.45.