Clasificarea tipurilor de flori folosind Machine Learning

Compararea clasificatorilor Random Forest, Naive Bayes și K-Nearest Neighbours

Tema 2

Tehnici de învățare automata

Proiect realizat de Marin Andreea- Iulia

Grupa 331AA

Facultatea de Automatică și Calculatoare

Universitatea Politehnică din București

Contents

1. Introducere

Scopul proiectului

În cadrul acestui proiect de machine learning, am investigat și comparat performanțele a trei clasificatori populare: Random Forest, Naive Bayes și K-Nearest Neighbors (KNN). Scopul principal al proiectului este de a identifica cel mai potrivit clasificator pentru o anumită problemă de clasificare, evaluându-i performanțele pe seturi de date de validare și testare.

Obiective

- Implementarea și Antrenarea Clasificatorilor: Am implementat și antrenat trei clasificatori de referință - Random Forest, Naive Bayes și KNN - pe setul nostru de date de antrenare.

- Evaluarea Performanțelor: Am evaluat performanțele clasificatorilor pe două seturi distincte de date: un set de validare pentru a ajusta parametrii și a preveni supraantrenarea, și un set de testare pentru a evalua abilitatea lor de generalizare.

- Compararea Rezultatelor: Am analizat rezultatele obținute de fiecare clasificator, comparând metrici precum acuratețea, raportul de clasificare și matricea de confuzie.

În continuare, vom detalia mai departe rezultatele, vom analiza factorii care influențează performanțele și vom extrage concluzii solide în privința alegerii celui mai potrivit clasificator pentru scopul nostru specific.

Descrierea succintă a modului de lucru

În cadrul proiectului meu de machine learning, am colectat și prelucrat datele pentru a clasifica între trei tipuri de flori: Random Forest, Naive Bayes și K-Nearest Neighbors (KNN). Am implementat și configurat acești clasificatori, ajustând parametrii pentru a obține cele mai bune rezultate. Prin antrenarea și evaluarea lor pe seturi distincte, am comparat performanțele, analizând atent factorii care au influențat rezultatele. Toate aceste informații sunt documentate în raportul meu, oferind o perspectivă clară și detaliată asupra comportamentului fiecărui clasificator.

2. Descrierea aplicației

Setul de date

Motivul Alegerii Setului de Date:

Am selectat un set de date cuprinzând două clase principale de flori: margarete și floarea-soarelui. Alegerea a fost făcută pentru a dezvolta un model de clasificare simplu și eficient, focalizat pe distingerea între aceste două tipuri de flori. Imaginile de train și validation aparțin unui set de date preluat de la adresa https://www.kaggle.com/datasets/imsparsh/flowers-dataset , verificat și împărțit manual, iar imaginile de test sunt preluate personal.

Structura și Organizarea:

Setul de date este împărțit în trei categorii principale:

/train: Conține imagini destinate antrenării modelului.

/validation: Imagini folosite pentru validarea și ajustarea modelului.

/test: Imagini utilizate pentru evaluarea finală a performanței modelului.

Clase de Flori:

- Daisy (Margarete):

Imagini cu margarete, evidențiind trăsăturile distinctive ale acestei flori.

Numele imaginilor sunt random etc.

- 567 imagini pentru setul de antrenare
- 107 imagini pentru setul de validare
- 42 imagini pentru setul de testare

- Sunflower (Floarea-soarelui):

Imagini cu florea-soarelui, evidențiind aspectele lor caracteristice.

Numele imaginilor sunt random etc.

- 543 imagini pentru setul de antrenare
- 105 imagini pentru setul de validare
- 49 imagini pentru setul de testare

Toate imaginile alese au extensia .jpg și rezoluții cuprinse intre 134-400px. Imaginile sunt redimensionate în interiorul programului la dimensiunea standard de (256,256).

Ierarhia folderelor:

Folderul principal de date /dataset este structurat astfel:

/dataset

/train

/daisy

daisy_image_1.jpg

daisy_image_2.jpg

...

/sunflower

sunflower_image_1.jpg

sunflower_image_2.jpg

...

/validation

(structură similară ca și în "train")

/test

(structură similară ca și în "train")

S-a folosit același set de date din Tema 1, acest proiect reprezentand continuarea acestuia. Prin aplicarea celor trei clasificatori pe același set de date, putem observa diferențele dintre aceștia.

Biblioteci utilizate

```
5   from sklearn.neighbors import KNeighborsClassifier
6   from sklearn.naive_bayes import GaussianNB
```

Pe lângă bibliotecile prezentate în cadrul primei teme, s-au folosit în plus:

- KNeighborsClassifier din scikit-learn :Am implementat și antrenat un model de clasificare folosind algoritmul K-Nearest Neighbors (KNN) din biblioteca scikit-learn. Acest algoritm s-a dovedit a fi un instrument eficient în rezolvarea problemei noastre de clasificare a imaginilor cu flori, bazându-se pe principiul că obiectele similare sunt adesea localizate în proximitatea spațială.
- GaussianNB din sklearn.naive_bayes: Am implementat și antrenat un model de clasificare folosind algoritmul Naive Bayes cu distribuție gaussiană (Gaussian Naive Bayes) din biblioteca scikit-learn. Acest algoritm a avut un rol semnificativ în sarcina noastră de clasificare a imaginilor cu flori, oferind o abordare eficientă și simplă, bazată pe asumpția naivă de independență între caracteristici.

Explicarea codului

Etapa de încarcare și pregatire a datelor a fost descrisă anterior, în cadrul Temei 1.

a) Antrenarea clasificatorilor

```
# Create and train the Random Forest Classifier
rf = RandomForestClassifier(n_estimators=100, random_state=42)
rf.fit(train_images, train_labels)

# Create and train the KNN Classifier
knn = KNeighborsClassifier(n_neighbors=5)
knn.fit(train_images, train_labels)

# Create and train the Naive Bayes Classifier
nb = GaussianNB()
nb.fit(train_images, train_labels)
```

- Se observă că numărul de vecini ales pentru evaluarea clasificatorului K-NN este 5, acesta fiind ales pentru certitudinea mai mare si simplificarea algoritmului.

b) Evaluarea pe setul de validare

```
# Set the threshold for Naive Bayes
threshold_nb = 0.5

# Make probability predictions on the validation set
rf_validation_pred_prob = rf.predict_proba(validation_images)
knn_validation_pred_prob = knn.predict_proba(validation_images)
nb_validation_pred_prob = nb.predict_proba(validation_images)

# Convert numeric labels back to class names
rf_predicted_class_names = rf.classes_[np.argmax(rf_validation_pred_prob, axis=1)]
knn_predicted_class_names = knn.classes_[np.argmax(knn_validation_pred_prob, axis=1)]
nb_predicted_class_names = nb.classes_[(nb_validation_pred_prob[:, 1] > threshold_nb).astype(int)]
```

- Pentru algoritmul K-NN s-a folosit aceeași abordare prezentată pentru clasificatorul Random Forest.
- Pentru algoritmul Naive Bayes s-a folosit o abordare bazată pe un ”threshold”. Mai precis, pentru un threshold de 0.5 s-a ales ca și clasă prezisă aceea care are procentul mai mare de 50%.

c) Evaluarea pentru setul de test

```
# Make probability predictions on the test set

rf_test_pred_prob = rf.predict_proba(test_images)
knn_test_pred_prob = knn.predict_proba(test_images)
nb_test_pred_prob = nb.predict_proba(test_images)

# Convert numeric labels back to class names
rf_test_predicted_class_names = rf.classes_[np.argmax(rf_test_pred_prob, axis=1)]
knn_test_predicted_class_names = knn.predict(test_images)
nb_test_predicted_class_names = nb.classes_[(nb_test_pred_prob[:, 1] > threshold_nb).astype(int)]
```

La fel ca la setul de validare.

d) Metrici de evaluare

```
# Display and print results for the validation set
# Random Forest
rf_validation_pred_labels = rf.classes_[np.argmax(rf_validation_pred_prob, axis=1)]
rf_validation_accuracy = accuracy_score(validation_labels, rf_validation_pred_labels)
rf_validation_classification_rep = classification_report(validation_labels, rf_validation_pred_labels)
rf_validation_conf_matrix = confusion_matrix(validation_labels, rf_validation_pred_labels)

print(f"Random Forest Validation Accuracy: {rf_validation_accuracy:.2%}")
print("Random Forest Validation Classification Report:")
print(rf_validation_classification_rep)
print("Random Forest Validation Confusion Matrix:")
print(rf_validation_conf_matrix)

# KNN
knn_validation_accuracy = accuracy_score(validation_labels, knn_predicted_class_names)
knn_validation_classification_rep = classification_report(validation_labels, knn_predicted_class_names)
knn_validation_conf_matrix = confusion_matrix(validation_labels, knn_predicted_class_names)

print(f"KNN Validation Accuracy: {knn_validation_accuracy:.2%}")
print("KNN Validation Classification Report:")
print(knn_validation_classification_rep)
print("KNN Validation Confusion Matrix:")
print(knn_validation_conf_matrix)

#Naive Bayes
nb_validation_pred_labels = nb.classes_[(nb_validation_pred_prob[:, 1] > threshold_nb).astype(int)]
nb_validation_accuracy = accuracy_score(validation_labels, nb_validation_pred_labels)
nb_validation_classification_rep = classification_report(validation_labels, nb_validation_pred_labels)
nb_validation_conf_matrix = confusion_matrix(validation_labels, nb_validation_pred_labels)

print(f"Naive Bayes Validation Accuracy: {nb_validation_accuracy:.2%}")
print("Naive Bayes Validation Classification Report:")
print(nb_validation_classification_rep)
print("Naive Bayes Validation Confusion Matrix:")
print(nb_validation_conf_matrix)
```

- Se calculează și se afișează acuratețea, raportul de clasificare și matricea de confuzie pentru fiecare clasificator în vederea evaluarii performanței modelului pe setul de test.

3. Interpretarea rezultatelor

Interpretarea metricilor de evaluare

- Clasificatorul Random Forest

  Acuratețea:

```
Random Forest Validation Accuracy: 83.02%
Random Forest Test Accuracy: 72.53%
```

  Pentru clasificatorul Random Forest s-a obținut cea mai mare acuratețe dintre cele 3, atât pe setul de validare cât și pe cel de testare. Aceasta acuratețe este datorată modului de decizie bazată pe arbori folosită de acest clasificator.

  Raportul de clasificare:

  Validare

```
Random Forest Validation Classification Report:
              precision    recall  f1-score   support

       daisy       0.84      0.82      0.83       107
   sunflower       0.82      0.84      0.83       105

    accuracy                           0.83       212
   macro avg       0.83      0.83      0.83       212
weighted avg       0.83      0.83      0.83       212
```

  Test

```
Random Forest Test Classification Report:
              precision    recall  f1-score   support

       daisy       0.65      0.88      0.75        42
   sunflower       0.85      0.59      0.70        49

    accuracy                           0.73        91
   macro avg       0.75      0.74      0.72        91
weighted avg       0.76      0.73      0.72        91
```

Observăm faptul că pentru setul de validare s-a obținut o precizie mai mare pentru clasa „Daisy”, în timp ce pentru setul de testare precizia a fost mai mare pentri clasa „Sunflower”. Aceasta diferență de precizie este datorată setului de date ales, ea indicând numărul de instanțe clasificate ca și „pozitiv” care sunt și corecte.

- Clasificatorul K-Nearest Neighbours

  Acuratețea:

```
KNN Validation Accuracy: 73.58%
KNN Test Accuracy: 58.24%
```

  Deși pentru setul de validare s-a obținut o acuratețe destul de bună, acuratețea setului de test reflectă faptul ca acest clasificator este mai slab decât clasificatorul Random Forest, deoarece se bazeaza doar pe asemănările între imaginea ce urmează a fi clasificată și imaginile de train. În momentul în care aceste similarități apar în mod neintenționat în imagini (fundal, culori etc), acest clasificator tinde spre o clasă ce în final, poate fi greșită. Numărul mic de vecini ales(5) duce la o certitudine mai mare că rezultatul va fi corect, de aici o acuratețe mai buna, datorită faptului ca există o șansă mai mare ca printre cei mai apropiați 5 vecini să se numere măcar 3 asignați clasei corecte din care face parte imaginea. Dacă, spre exemplu, am fi ales un număr de 20 de vecini, certitudinea că măcar 10 dintre aceștia sunt corecți ar fi fost mai mică, datorită ariei mai mare de căutare.

  Raportul de clasificare:

  Validare

```
KNN Validation Classification Report:
              precision    recall  f1-score   support

       daisy       0.71      0.80      0.75       107
   sunflower       0.77      0.67      0.71       105

    accuracy                           0.74       212
   macro avg       0.74      0.74      0.73       212
weighted avg       0.74      0.74      0.73       212
```

Test

KNN Test Classification Report:

	precision	recall	f1-score	support
daisy	0.53	0.81	0.64	42
sunflower	0.70	0.39	0.50	49
accuracy			0.58	91
macro avg	0.62	0.60	0.57	91
weighted avg	0.62	0.58	0.57	91

Se observă o discrepanță mare între precizia algoritmului pentru clasa „Margarete” în cadrul setului de validare și precizia pentru aceeași clasă în cadrul setului de test. Această precizie(precision = TP/(TP+FP)) scăzută indică faptul ca există mai multe instanțe clasificate fals-pozitiv în cadrul setului de test.

În același timp, se observă un indice de recall constant mai mare pentru clasa „Sunflower” .(recall=TP/(TP+FN)). Acesta indică faptul ca din numărul total de instanțe real-pozitive, mai multe au fost clasificate corect(drept TP) în cadrul clasei „sunflower” decât „daisy”.

- Clasificatorul Naive Bayes

Acuratețea

```
Naive Bayes Validation Accuracy: 80.19%
Naive Bayes Test Accuracy: 65.93%
```

Deși este considerat un clasificator „naiv” datorită modului în care ia deciziile, bazându-se pe ipoteză ca imagile au caracteristici complet independente, Naive Bayes reușește să livreze o acuratețe mai bună decât K-NN, atât pe setul de validare cât și pe cel de testare. Considerând faptul că efectuăm clasificarea pe două tipuri de flori care au deosebiri majore vizual, abordarea acestui clasificator pare a fi destul de utilă, Astfel, el poate considera ca și caracteristici independente:

- Culoarea petalelor
  Margaretele au (predominant) petale albe, în timp ce floarea- soarelui are petale galbene/

- Dimensiunea petalelor
  Margaretele au petale mai scurte în comparație cu floarea-soarelui.

- Discul floral

  Margaretele au discul floral galben, în timp ce floarea-soarelui îl are maro închis-negru.

  Deși aceste caracteristici par definitorii pentru cele două tipuri de flori, există diferiți factori ce pot influența modul în care caracteristicile sunt percepute și duc la o clasificare greșită.

  Spre exemplu, o poză întunecată poate face ca discul floral al margaretei să pară mai închis la culoare(similar florii-soarelui). Când clasificatorul Naive-Bayes face decizia pe baza culorii discului floral, acesta va încadra floarea margaretă în clasa „Sunflower”, doar datorită caracteristicii ce este vizualizată ca fiind independentă și decisivă pentru atribuirea clasei.

  Similar, o imagine mai luminoasă poate duce la încadrarea greșită a florii „Sunflower” în clasa „Daisy”(dpdv al culorii discului floral), distorsionarea imaginii datorită lentilei și unghiului de fotografiere poate duce la alungirea petalelor margaretei sau scurtarea petalelor florii soarelui. De asemenea, lumina caldă (provenită, spre exemplu, de la apusul soarelui) poate duce la impresia de petale galbene la floarea margaretă, ce rezultă într-o clasificare greșită.

  Raportul de clasificare:

  Validare

Naive Bayes Validation Classification Report:

	precision	recall	f1-score	support
daisy	0.81	0.79	0.80	107
sunflower	0.79	0.81	0.80	105
accuracy			0.80	212
macro avg	0.80	0.80	0.80	212
weighted avg	0.80	0.80	0.80	212

  Test

Naive Bayes Test Classification Report:

	precision	recall	f1-score	support
daisy	0.60	0.81	0.69	42
sunflower	0.76	0.53	0.63	49
accuracy			0.66	91
macro avg	0.68	0.67	0.66	91
weighted avg	0.69	0.66	0.65	91

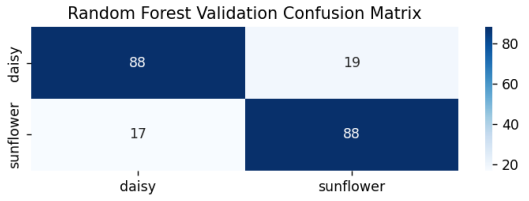
  Pentru setul de validare, s-au obținut valori apropiate atât pentru precizie cât și pentru indicele de recall. Aceste valori conduc la concluzia că, deși Naive Bayes

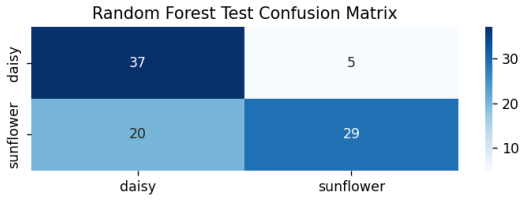
se bazează pe independența caracteristicilor, a avut o aplicabilitate bună asupra setului de validare (caracteristicile florilor din imagini sunt bine definite, există mai puține cazuri în care factorii externi distorsionează aceste caracteristici).

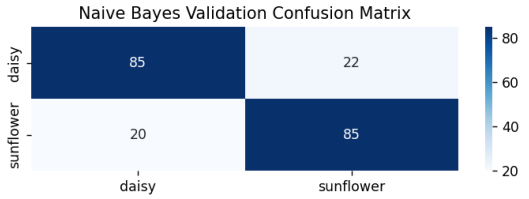
Pe de altă parte, pentru setul de testare s-au obținut valori vizibil mai bune ale indicelui de recall pentru clasa daisy în comparație cu sunflower. Acesta indică faptul că, din toate instanțele de real-pozitiv ale clasei daisy, au existat mai puține cazuri de fals-negativ(daisy clasificată ca și sunflower) decât în cazul clasei sunflower(au fost mult mai multe cazuri de fals negative- sunflower clasificată ca și daisy). Cum am menționat anterior, aceste cazuri de fals negativ pot fi datorate calității imaginilor, factorul de recall mic sugerând faptul ca un procent mare de imagini ale clasei sunflower prezintă caracteristici extrem de similare cu o floare daisy.

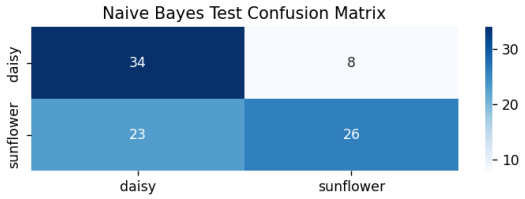
Interpretarea figurilor

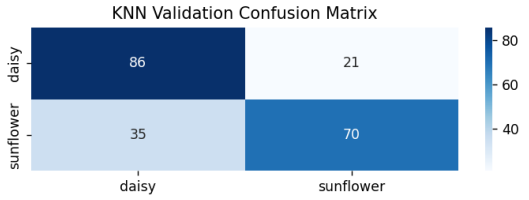
Reprezentarea matricelor de confuzie

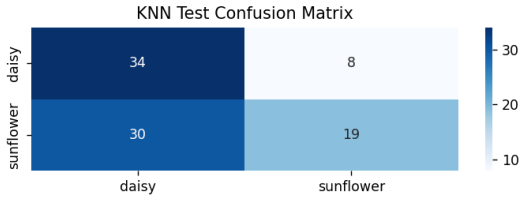












Comparăm, pe rând, fiecare cadran al matricelor de confuzie:

- Cazul true-positive

  True positive(TP) este cazul în care o instanță a fost clasificată drept pozitiv și ea este, de fapt, pozitivă.
  Pentru exemplul nostru, vom considera pozitivul ca fiind clasa „Daisy”.

  Pentru setul de validare, cele mai multe instanțe TP au fost găsite de către algoritmul Random Forest(88), în timp ce cele mai puține au fost rezultate din algoritmul Naive Bayes(85). Deși toate clasificatoarele au obținut un număr ridicat de instanțe „daisy” clasificate corect, Naive Bayes se pare că a confundat cel mai des floarea daisy cu floarea sunflower, rezultând în mărirea numarului de instanțe fals negative(FN).

  Pentru setul de testare se observă aceeași ordine a clasificatorilor, numărul cel mai mare de clasificări corecte fiind dat de Random Forest(37), în timp ce K-NN și Naive Bayes au obținut același rezultat(34).

- Cazul false-negative

  False negative(FN) se referă la numărul instanțelor pozitive ce au fost clasificate (greșit) drept negative(daisy a fost clasificată ca și sunflower).
  Numărul cel mai mare de instanțe clasificate fals-negative în setul de validare a fost obținut de clasificatorul K-NN, ce se pare că a confundat cel mai des floarea daisy cu floarea sunflower(22 instanțe). Cel mai mic număr de FN a fost obținut de Random Forest(19 instanțe).
  Pentru setul de testare, Naive Bayes și K-NN au obținut din nou același număr de instanțe, de data aceasta clasificate fals-negativ(8 instanțe). Random Forest a avut un număr mai mic de instanțe FN (5).

- Cazul false-positive

  False positive(FP) înseamnă numărul instanțelor negative clasificate greșit drept pozitive (sunflower a fost clasificată drept daisy).

Pentru setul de validare, cele mai multe instanțe de sunflower clasificate greșit drept daisy a fost obținut de către clasificatorul K-NN(35), indicând faptul că unele imagini cu sunflower prezintă mai multe similarități față de daisy decât față de propria clasă (a găsit 3 sau mai mulți vecini apropiați aparținând clasei daisy, din 5 vecini luați în considerare). Cele mai puține greșeli de tipul FP au fost realizate tot de către clasificatorul Random Forest(17).
Pentru setul de testare, se observă că tot clasificatorul K-NN a produs cele mai multe confuzii între sunflower și daisy, un număr de 30. Random Forest a clasificat doar 20 de imagini cu sunflower drept daisy.

- Cazul true-negative

True negative(TN): instanțe negative clasificate drept negative.
Pentru setul de validare, cele mai multe instanțe de sunflower clasificate corect a fost obținut tot de clasificatorul Random Forest(88), iar cele mai puține de clasificatorul K-NN(70). Din moment cel K-NN a fost clasificatorul care a confundat cel mai des sunflower cu daisy, era de așteptat faptul că va avea cele mai puține instanțe clasificate corect drept sunflower, deoarece a avut un număr mai mare de instanțe negative pe care le-a clasificat greșit.
Pentru setul de testare se obțin, desigur, valori ridicate pentru TN la Random Forest(29) și scăzute pentru K-NN.

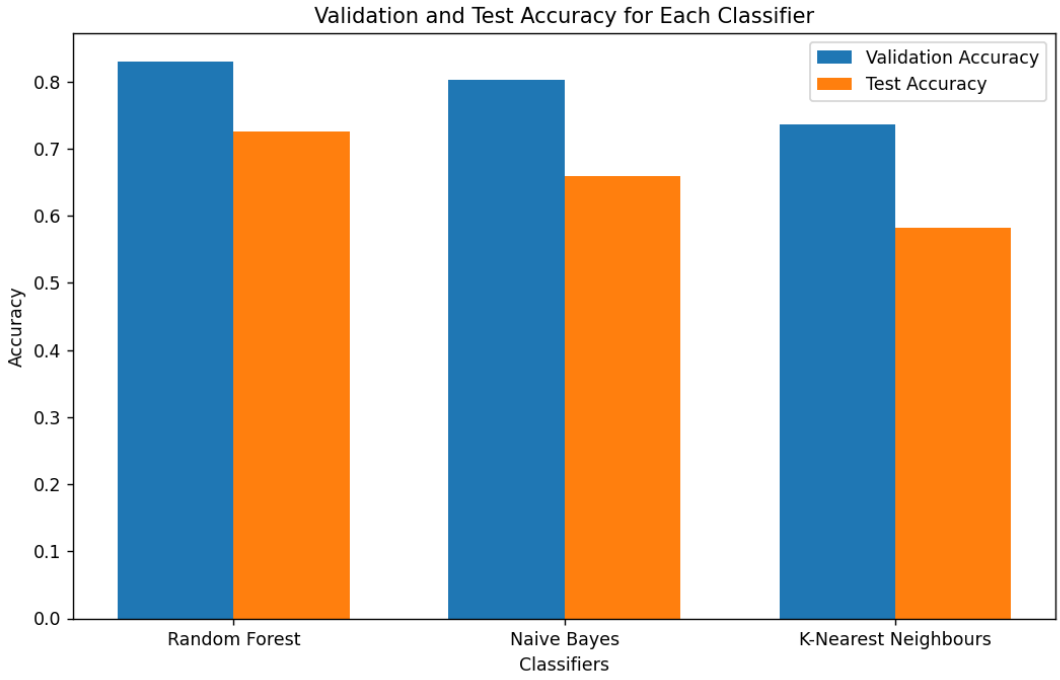
Din matricile de confuzie, putem spune cu certitudine că Random Forest este cel mai bun clasificator pentru problema aleasă.

De altfel, se observă faptul că Naive Bayes a confundat mai multe instanțe de daisy, clasificându-le greșit drept sunflower (mai multe cazuri de false negative), în timp ce K-NN a avut mai multe instanțe de false-positive (a clasificat instanțe de sunflower drept daisy). Acest pattern de repetare a greșelilor este, de fapt, datorat modului în care cei doi clasificatori iau deciziile. Naive Bayes a confundat de mai multe ori Daisy cu Sunflower, probabil datorită faptului că, uneori, margaretele au petalele galbene, caracteristică care face ca o floare să fie atribuită automat clasei sunflower- tratarea caracteristicilor ca fiind independente- orice floare cu petale galbene este cu siguranță floarea-soarelui.

De asemenea, numărul mare de FP obținut de K-NN sugerează faptul că pentru 35, respectiv 30 de imagini cu floarea daisy, s-au găsit mai mulți vecini din cei 5 cei mai apropiați care aparțineau clasei sunflower.

Acest lucru poate fi datorat mai multor factori și este mai greu de găsit cauza problemei, dar putem spune, în principiu, că anumite imagini cu margarete prezintă mai multe caracteristici generale ce se găsesc si în imaginile cu floarea-soarelui. Spre exemplu, poate a fost pozat un câmp plin cu margarete, confuzia fiind datorată faptului că modelul a fost antrenat pe imagini cu o margaretă sau un buchet. Câmpul predominat de flori sau prezența cerului pe fundalul imaginii poate duce la falsa impresie de floarea soarelui, mai ales dacă imaginea este neclară si culoarile petalelor și a discului floral nu sunt vizibile complet.

Diagrama acurateței



S-a realizat o diagramă în care sunt figurate, sub formă de coloane, valorile acurateței celor trei clasificatori, atât pe setul de validare cât și pe cel de testare.

Se observă că fiecare clasificator a obținut o acuratețe mai mică pe setul de testare decât pe cel de validare, lucru datorat(posibil) numărului mai mic de imagini din setul de testare.

Overall, cea mai bună performanță a fost oferită de clasificatorul Random Forest, în timp ce cea mai slabă este a clasificatorului K-Nearest Neighbours.

4. Exemple

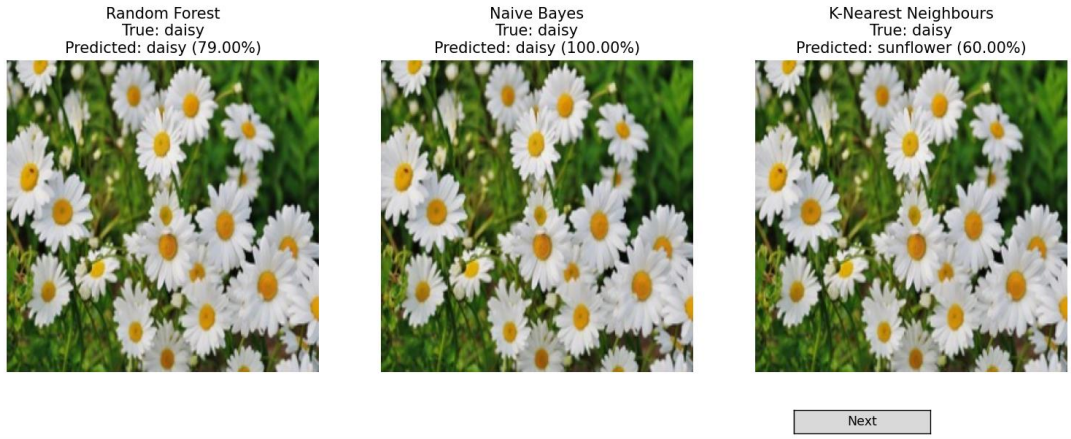
Se observă o clasificare de tip true-positive, realizată corect de către toți clasificatorii.

Vizualizând procentele de apartență la clasă, observăm ca Random Forest, prin deciziile multiple realizate sub formă de arbori, este capabil să obțina procente random(79%), arătând faptul ca a gasit similarități și cu clasa Sunflower(de 21%), însă procentul mai mare către clasa Daisy a făcut ca decizia finală să fie apartenența la clasa Daisy.

Pentru Naive Bayes, singurul procent obținut este 100%. Acest lucru este datorat modului în care privește caracteristicile (drept individuale). Mai explicit, nu este capabil să găsească similarități cu mai mult de o clasă. Culoarea petalelor margaretelor din imagini (roz și alb) a reprezentat factorul decisiv al clasificării, Naive Bayes fiind sigur că doar margaretele pot avea această culoare a petalelor, atribuind orbește procentul de 100% clasei Daisy, neverificând și alte atribute(cum ar fi culoarea discului floral, care pentru anumite flori din imagine apare mai închisă, similar cu floarea soarelui).

K-NN poate obține doar procente de 0%,20%,40%,60%,80% și 100%, datorită numărului de vecini(5) după care se ia decizia. În cazul prezentat, procentul 80% indică faptul că, din cei 5 vecini cei mai apropiați, 4 dintre aceștia aparțin clasei „Daisy”, rezultând într-o clasificare corectă.

Alte exemple:



Random Forest, Naive Bayes -TP

K-NN – FN

Random Forest, K-NN -TP

Naive Bayes- FN