Corcodel Vladut-Ionut
334AA

TEMA 2 TIA

CLASIFICARE MERE SI BANANE UTILIZAND ALGORITMII K-NEAREST NEIGHBORS SI NAIVE BAYES

1.Prezentarea temei:

Tema se axeaza pe clasificarea automata a imaginilor in doua categorii, MERE și BANANE, utilizand algoritmii de invatare automata KNN și NB. In urma aplicarii acestor doi algoritmi, vom compara performantele lor pe acelasi set de date, dar si cu rezultatele algoritmului k-means utilizat la tema1 pentru a evidentia diferentele si pentru a determina care este mai eficient din punct de vedere al acuratetii predictiilor.

2.Prezentarea setului de date si a etapei de preprocesare:

Setul de date este impartit in 3 foldere primul de antrenare, al doilea de validare si al treilea de testare:

1)Antrenare: se numeste dataset si contine 2 subfoldere apple si banana ce contin poze cu mere si banane (in proportie de 70%)

2)Validare: se numeste validation si contine 2 subfoldere apple si banana ce contin poze cu mere si banana (in proportie de 15-20%)

3)Testare: se numeste test si contine 2 subfoldere apple si banana ce contin poze cu mere si banana (in proportie de 10%)

In ambii algoritmi este important sa redimensionez pozele (100x100), deoarece setul de date a fost creat din extragerea imaginilor din diferite seturi de date de pe internet si alipirea acestora pentru al crea. Pe langa extragere a fost nevoie sa tin cont si de proportiile indicate pentru ca setul meu de date sa fie valid, acetea fiind prezentate mai sus.

Corcodel Vladut-Ionut
334AA

3.Descriere algoritmi si parametrii utilizati:

KNN:

-Imaginile sunt redimensionate la 100x100 si liniarizate facandu-le adecvate pentru prelucrarea cu knn.

-Scalarea datelor utilizand functia ‘StandardScaler’ inainte de aplicarea knn, deoarece acesta este sensibil la scara caracteristicelor (asa asigura ca fiecare caracteristica contribuie egal la calculul distantei)

-Alegerea valorilor K(numarul de vecini)

- Pentru o instanta necunoscuta algoritmul calculeaza distanta dintre acest punct si toate celelalte puncte din setul de antrenament.

- Dupa calcularea distantelor, se identifica cei mai apropiați K vecini ai punctului de date necunoscut. In cazul nostru fiind 1

-Odata identificați, vecinii „voteaza” pentru eticheta lor. Clasa care apare cel mai frecvent în randul celor K vecini este atribuita punctului de date necunoscut.

Parametrii:

Definirea grilei de hiperparametri:

```
param_grid = {'kneighborsclassifier__n_neighbors': [1, 3, 5, 7, 9]}
```

GridSearchCV este inițializat cu modelul K-NN, grila de hiperparametri și alte parametri precum cv pentru numărul de folduri în cross-validation și scoring pentru metrica de evaluare.

```
# Inițializarea GridSearchCV
grid_search = GridSearchCV(knn_model, param_grid, cv=5, scoring='accuracy')
```

X_train= tablou care conține caracteristicile sau atributele (features) ale datelor de antrenare

Y_train=Este un vector care conține etichetele corespunzătoare fiecărui exemplu din setul de antrenare.

```
(X_train, y_train)
```

Afisarea celui mai bun hiperparametru:

Corcodel Vladut-Ionut
334AA

```
Best Hyperparameters: {'kneighborsclassifier__n_neighbors': 1}
```

NB:

-In cazul meu teorema lui NB este folosita pentru a calcula probabilitatea ca o imagine apartine clasei ‘mar sau ‘banana’ pe baza pixelilor imaginii

- Algoritmul Naive Bayes face presupunerea ca toate caracteristicile (în cazul dvs., pixelii imaginii) sunt independente una fata de cealalta. Aceasta înseamna ca probabilitatea sa fie un anumit pixel de o anumita valoare în imagine nu este influențata de prezenta sau absenta altor pixeli în aceeasi imagine

- Calcularea produsului probabilitatii de verosimilitate (probabilitatea de a observa pixelii din imagine dat fiind clasa ‘mar sau ‘banana’) pentru fiecare pixel din imagine și probabilitatea anterioara (probabilitatea generala a clasei ‘mar sau ‘banana’ în setul de date)

-Dupa calcularea probabilitatilor pentru ambele clase, algoritmul alege clasa cu cea mai mare probabilitate. Cu alte cuvinte, decide ca imaginea apartine clasei ‘mar sau ‘banana’pe baza probabilitatilor calculate.

Parametrii:

Definiți grila de hiperparametri pentru căutarea în grilă cu GridSearchCV. În acest caz, se ajustează parametrul alpha al clasificatorului MultinomialNB.

```
# Definirea hiperparametrilor pe care vrei să-i ajustezi
param_grid = {
    'alpha': [0.1, 1.0, 10.0],  # Exemplu: hiperparametru alpha pentru clasificatorul MultinomialNB
}
```

Inițializezi un clasificator MultinomialNB și apoi un obiect GridSearchCV pentru a căuta cei mai buni hiperparametri folosind validarea încrucișată (cv=5).

```
clf = MultinomialNB()
grid_search = GridSearchCV(clf, param_grid, cv=5, scoring='accuracy')
grid_search.fit(X_train, y_train)
```

După căutarea în grilă, antrenezi clasificatorul cu cei mai buni hiperparametri găsiți.

```
best_clf = grid_search.best_estimator_
```

Corcodel Vladut-Ionut
334AA

K-Means:

-Numărul de clustere (n_cluster): 2

-Random state: 0

-Numărul de inițieri (n_init): 20

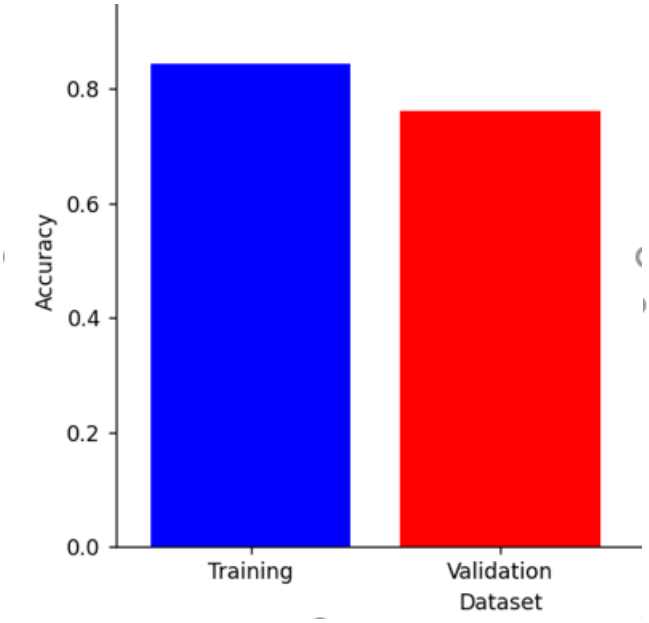
-Numărul maxim de iterații (max_iter): 300

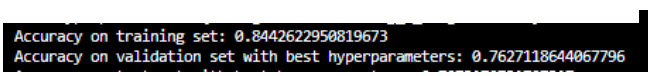
4.Proceslui de antrenare si validare:

-Antrenarea Clasificatorului KNN cu Cei Mai Buni Hiperparametri (hiperparametrii sunt ajustati in urma antrenarii si testarii pe setut de validare pentru a obtine rezultate mai bune)

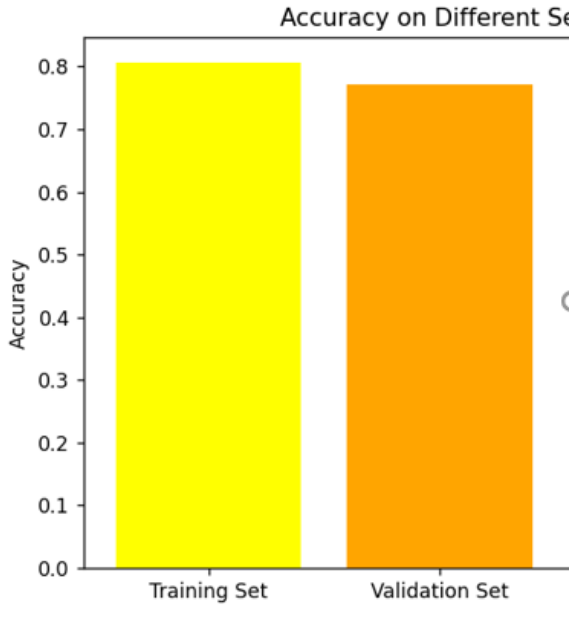
-In urma antrenarii se observa in graficele urmatoare acuratetea pe seturile de training si validation

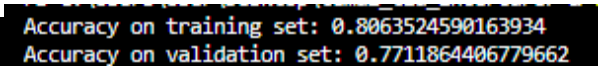
Algoitm knn





Algoritm NB





Asta insemna ca in cazul antrenarii a prezis clasa corect doar 84,4% poze corect(knn).

Asta insemna ca in cazul antrenarii a prezis clasa corect doar 80,6% poze corect(nb).

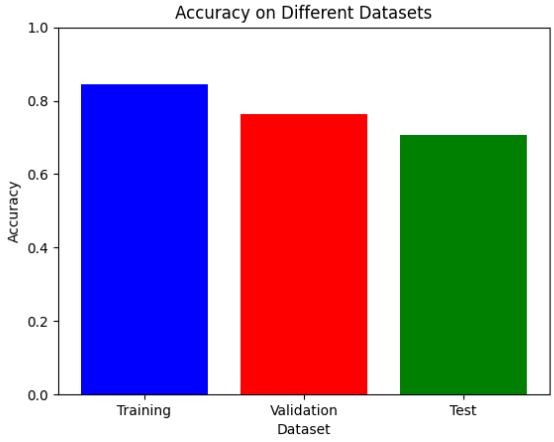
Corcodel Vladut-Ionut
334AA

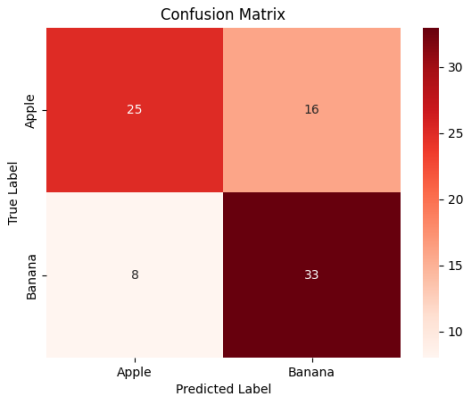
5. Procesul de testare:

Desi knn si nb folosesc diferitete metode de predictie(nb- alege clasa in urma probabilitatii calculate si knn- ‘vecinii’ aleg clasa imaginilor noi, dar ambele dupa antrenare sunt testate pe setul de validare pentru a ajusta parametrii care ajuta la selectia clasei), ambele urmaresc sa aiba o acuratete cat mai mare a identificarii claselor ‘mar’ sau ‘banana’.

In urma algoritmului **knn** s-au obtinut urmatoarele rezultate pentru testare:

-acuratete pe testare=0.7073170731707317





Pe diagonala principala sunt valorile corect clasificate:

25 de imagini cu mere au fost corect clasificate ca mere.

33 de imagini cu banane au fost corect clasificate ca banane.

Pe diagonala secundara sunt valorile eronate:

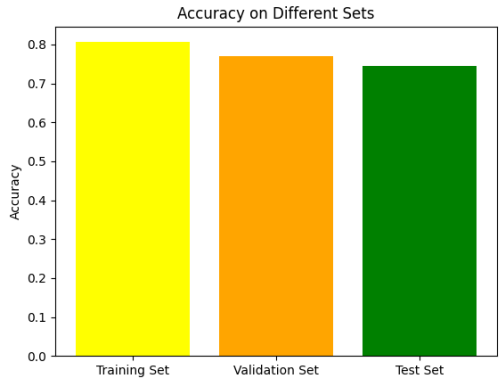
16 de imagini cu mere au fost clasificate greșit ca banane.

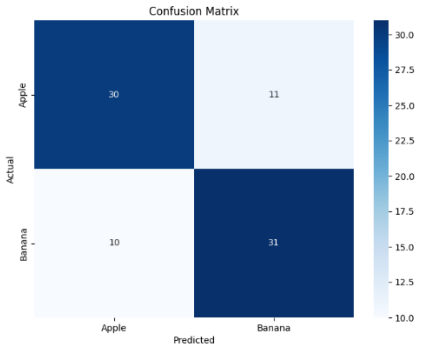
8 de imagini cu banane au fost clasificate greșit ca mere.

In urma algoritmului **NB** s-au obtinut urmatoarele rezultate pentru testare:

-acuratete pe testare= 0.7439024390243902

Corcodel Vladut-Ionut
334AA





Pe diagonala principala sunt valorile corect clasificate:

30 de imagini cu mere au fost corect clasificate ca mere.

31 de imagini cu banane au fost corect clasificate ca banane.

Pe diagonala secundara sunt valorile eronate:

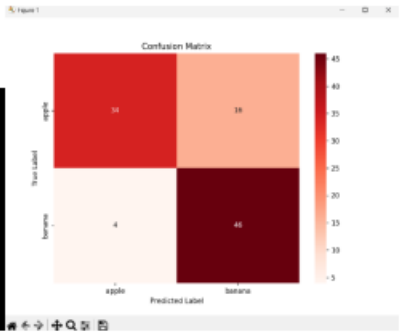
11 de imagini cu mere au fost clasificate greșit ca banane.

10 de imagini cu banane au fost clasificate greșit ca mere.

In urma algoritmului **K-means** s-au obtinut urmatoarele rezultate pentru testare:

Classification Report on TEST set:

	precision	recall	f1-score	support
apple	0.89	0.68	0.77	50
banana	0.74	0.92	0.82	50
accuracy			0.80	100
macro avg	0.82	0.80	0.80	100
weighted avg	0.82	0.80	0.80	100



- 0.80 procentul total de predicții corecte.

<u>Dupa cum se poate vedea si din matricea de confuzie:</u>

-Pentru clasa "apple", 34 de exemple au fost corect clasificate (True Negatives), iar 16 au fost clasificate greșit ca "banana" (False Positives).

Corcodel Vladut-Ionut
334AA

-Pentru clasa "banana", 46 de exemple au fost corect clasificate (True Positives), iar 4 au fost clasificate greșit ca "apple" (False Negatives)

6.Avantaje si dezavantaje:

Avantaje:- Simplicitate si usurinta in implementare: Atat KNN cat si Naive Bayes sunt modele relativ simple și directe. Aceste modele sunt usor de implementat si de inteles, facandu-le potrivite pentru probleme de clasificare de baza.

-Eficienta în timp real: KNN, in special, poate fi eficient in scenarii unde decizia trebuie luata rapid.

Dezavantaje: -Performanta limitata pe date complexe sau de mare dimensiune: KNN poate deveni ineficient pe masura ce dimensiunea datelor creste, iar Naive Bayes poate avea performante slabe pe date cu relatii complexe sau caracteristici interdependente.

-Sensibilitate la date neuniforme si zgomotoase: ambele modele pot avea performante reduse în cazul datelor cu zgomot sau anomalii.

-Necesitatea selectarii parametrilor potriviti: Alegerea numarului de vecini în KNN si a parametrilor modelului in Naive Bayes poate influenta semnificativ performanta.

7.Concluzii:

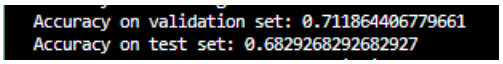
-Din punct de vedere al acuratetei pe testare algoritmul Naive Bayes este mai eficient pe acest set de date decat KNN, dar ambii algoritmi au rezultate bune ambii avand performanta de peste 70% pe seturi.

-Algoritmul k-MEANS este mai eficient decat acesti 2 algoritmi, deoarece a fost folosit grayscale-ul in preprocesarea imaginii cee ace-l ajuta la antrenare si din cauza ca setul de date are grupuri evidente si ii este mai usor sa-si defineasca clusterele.

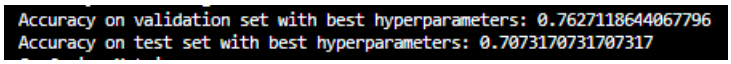
- KNN și Naive Bayes au avut performanțe variabile pe seturile de antrenare, validare și testare. Aceasta evidentiaza importanta selectiei atente a hiperparametrilor și a preprocesarii datelor pentru optimizarea performantei.

-Este importanta ajustarea parametrilor in urma testarii setului de validare, deoarece creste acuratetea prezicerilor pe testare si validare, exemplu pe algoritmul knn:

FARA AJUSTARE



CU AJUSTARE



Corcodel Vladut-Ionut
334AA

8.Bibliografie:

-Biblioteci: sklearn.model_selection.GridSearchCV, sklearn.neighbors.KNeighborsClassifier, sklearn.metrics, sklearn.preprocessing.StandardScaler, sklearn.pipeline.make_pipeline, PIL, numpy, os, matplotlib.pyplot, seaborn;

-Setul de date a fost obtinut prin alipirea mai multor seturi de date:

- https://www.kaggle.com/

- https://roboflow.com/

- https://www.google.com/imghp?hl=en