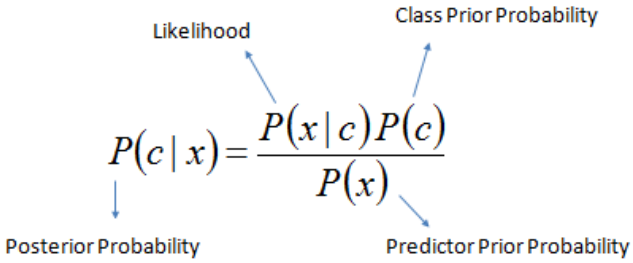
Naive Bayes

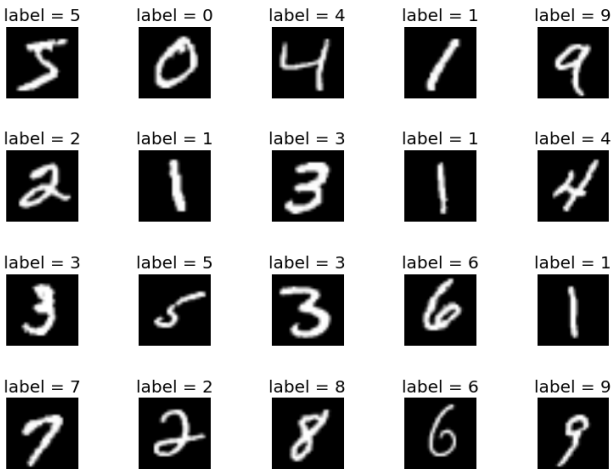


Regula Bayes

În acest laborator vom clasifica cifrele scrise de mână din subsetul **MNIST** folosind Naive Bayes.

MNIST[1] este o bază de date cu cifre scrise de mână (0-9), conținând 60.000 de imagini pentru antrenare și 10.000 pentru testare. Imaginile sunt alb-negru având dimensiunea de 28x28 pixeli. În cadrul laboratorului vom lucra pe un subset, împărțit astfel:

- ➔ În 'train_images.txt' sunt 1.000 de imagini din mulțimea de antrenare, fiecare fiind stocată pe câte o linie a matricei de dimensiune 1000 x 784 (28 x 28 = 784).
- ➔ În 'test_images.txt' sunt 500 de imagini din setul de testare.
- ➔ Fișierele 'train_labels.txt' și 'test_labels.txt' conțin etichetele imaginilor.



Exemple de imagini din setul de date MNIST.

Descărcați arhiva care conține datele de antrenare și testare de aici.

[1] http://yann.lecun.com/exdb/mnist/

Pentru vizualizarea unei imagini din mulțimea de antrenare trebuie să redimensionăm vectorul de 1 x 784 la 28 x 28.

```
import numpy as np
import matplotlib.pyplot as plt

train_images = np.loadtxt('train_images.txt') # incarcam imaginile
train_labels = np.loadtxt('train_labels.txt', 'int')  # incarcam etichetele avand
                                                      # tipul de date int
image = train_images[0, :] # prima imagine
image = np.reshape(image, (28, 28))
plt.imshow(image.astype(np.uint8), cmap='gray')
plt.show()
```

Deoarece datele noastre (valorile pixelilor) sunt valori continue, va trebui sa le transformăm în valori discrete cu ajutorul unei histograme. Vom stabili numărul de intervale la care vom împărți lungimea intervalului valorilor continue, apoi vom asigna fiecărei valori continue indicele intervalul corespunzător.

```
bins = np.linspace(start=0, stop=255, num=num_bins)  # returneaza intervalele
x_to_bins = np.digitize(x, bins) # returneaza pentru fiecare element intervalul
                                 # corespunzator
                                 # Atentie! In cazul nostru indexarea elementelor va
                                 # incepe de la 1, intrucat nu avem valori < 0
```

1. Antrenarea clasificatorului (fit)

O imagine $X = \{x_1, x_2, \ldots, x_{784}\}$ din mulțimea de antrenare are dimensiunea de 1x784. Conform presupunerii clasificatorului Naive Bayes, vom considera fiecare pixel ca fiind un atribut ***independent*** în calcularea probabilității apartenenței lui *X* la clasa *c*.

$$P(c|X) = p(c)\prod_{i=1}^{784} P(x_i|c) \quad | \; aplicăm\ logaritm$$

$$log(P(c|X)) = log(P(c)) + \sum_{i=1}^{784} log(P(x_i|c))$$

Pentru aplicarea regulii Naive Bayes avem nevoie de:

1. $P(c) = \frac{numărul\ exemplelor\ din\ clasa\ c}{numărul\ total\ de\ exemple}$, *probabilitatea ca un exemplu să se afle în clasa c*
2. $P(x|c) = \frac{numărul\ exemplelor\ din\ clasa\ c\ care\ sunt\ egale\ cu\ x}{numărul\ exemplelor\ din\ clasa\ c}$, *probabilitatea de a avea atributul x în clasa c*

2. Prezicerea etichetelor pe baza clasificatorul (predict)

$$P(c|X) = p(c)\prod_{i=1}^{n} P(x_i|c)\,, unde\ X = \{x_1, x_2, \ldots, x_n\}\ cu\ x_1, \ldots, x_n\ atribute\ independente.$$

Probabilitatea ca exemplul $X = \{x_1, x_2, \ldots, x_{784}\}$ să fie în clasa c, se obține prin înmulțirea (sau adunarea logaritmilor) probabilităților individuale ale atributelor acestuia condiționate de clasa c. Vom calcula P(c|X) pentru fiecare clasă c (c∈ [1, num_classes]), iar eticheta finală este dată de clasa cu probabilitatea cea mai mare.

Biblioteca Scikit-learn

În continuare vom folosi biblioteca ***Scikit-learn***. Aceasta este dezvoltată în Python, fiind integrată cu NumPy și pune la dispoziție o serie de algoritmi optimizați pentru probleme de clasificare, regresie și clusterizare.

Pas 1: Instalarea librăriei
```
pip install scikit-learn
```
Pas 2: Importarea modelului
```
from sklearn.naive_bayes import MultinomialNB
```
Pas 3: Definirea modelului
```
naive_bayes_model = MultinomialNB()
```
Pas 4: Antrenarea modelului
```
naive_bayes_model.fit(training_data, training_labels)
```
Pas 5.1: Prezicerea etichetelor
```
naive_bayes_model.predict(testing_data)
```
Pas 5.2: Calcularea acurateții
```
naive_bayes_model.score(testing_data, testing_labels)
```

Exerciții

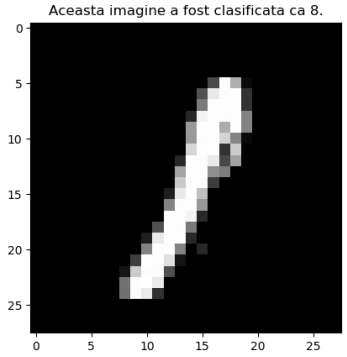
1. Se dă mulțimea de antrenare, reprezentând înălțimea în cm a unei persoane și eticheta corespunzătoare:
   [(160, F), (165, F), (155, F), (172, F), (175, B), (180, B), (177, B), (190, B)].
   Împărțind valorile continue (înălțimea) în 4 intervale (150-160, 161-170, 171-180, 181-190), calculați probabilitatea ca o persoană având 178 cm, să fie fată sau să fie băiat, folosind regula lui Bayes.

2. Știind că valoarea minimă a unui pixel este *0*, iar valoarea maximă este *255,* calculați capetele a *num_bins* intervale (utilizați funcția linspace). Definiți metoda *values_to_bins* care primește o matrice de dimensiune (n_samples, n_features) și capetele intervalelor calculate anterior, iar pentru fiecare

exemplu și fiecare atribut calculează indexul intervalului corespunzător (utilizați funcția np.digitize).
Folosiți funcția definită pentru a discretiza mulțimea de antrenare și cea de testare.

3. Calculați acuratețea pe *mulțimea de testare* a clasificatorul Multinomial Naive Bayes, împărținând intervalul pixelilor în 5 sub-intervale.
   OBS. Acuratețea pe care trebuie să o obțineți pentru num_bins = 5 este de 83.6%.

4. Testați clasificatorul Multinomial Naive Bayes pe subsetul MNIST folosind$num_bins \in \{3, 5, 7, 9, 11\}$.
5. Folosind numărul de sub-intervale care obține cea mai bună acuratețe la exercițiul anterior, afișați cel puțin 10 exemple misclasate.



6. Definiți metoda *confusion_matrix(y_true, y_pred)* care calculează matricea de confuzie. Calculați matricea de confuzie folosind predicțiile clasificatorului anterior.
   Obs:
   - Matrice de confuzie $C = c_{ij}$, numărul exemplelor din clasa *i* care au fost clasificata ca fiind în clasa *j*.

Clasa actuală↓ Clasa prezisă →	1	2	3
1	Nr. exemplelor din clasa 1 care au fost clasificate ca fiind in clasa 1	Nr. exemplelor din clasa 1 care au fost clasificate ca fiind in clasa 2	Nr. exemplelor din clasa 1 care au fost clasificate ca fiind in clasa 3
2	Nr. exemplelor din clasa 2 care au fost clasificate ca fiind in clasa 1	Nr. exemplelor din clasa 2 care au fost clasificate ca fiind in clasa 2	Nr. exemplelor din clasa 2 care au fost clasificate ca fiind in clasa 3

3	Nr. exemplelor din clasa 3 care au fost clasificate ca fiind in clasa 1	Nr. exemplelor din clasa 3 care au fost clasificate ca fiind in clasa 2	Nr. exemplelor din clasa 3 care au fost clasificate ca fiind in clasa 3

- Matricea de confuzie pentru clasificatorul anterior este:

```
[[51.  0.  0.  0.  0.  0.  1.  0.  1.  0.]
 [ 0. 48.  0.  0.  0.  0.  0.  0.  4.  0.]
 [ 2.  1. 50.  1.  1.  0.  1.  1.  1.  0.]
 [ 0.  0.  1. 49.  0.  0.  0.  0.  0.  3.]
 [ 0.  0.  0.  0. 33.  0.  0.  0.  2. 11.]
 [ 1.  0.  0.  9.  0. 34.  1.  0.  6.  1.]
 [ 1.  1.  0.  0.  1.  0. 43.  0.  2.  0.]
 [ 0.  1.  0.  0.  2.  0.  0. 41.  0.  6.]
 [ 0.  1.  3.  3.  1.  1.  1.  1. 34.  1.]
 [ 0.  0.  1.  1.  5.  0.  0.  0.  0. 35.]]
```