task1:

-casez matricea imaginii ‘photo’ la double

-aplic functia SVD asupra photo si obtin matricele U S V, unde A = U * S * V’

-din U pastrez doar primele k coloane

-din S pastrez primele k linii si primele k coloane

- calculez aproximatia imaginii initiale folosind matricele reduse calculate mai sus: new_photo = U * S *V’

- castez new_photo la unit8

task2:

- castez matricea imaginii ‘photo’ la double

- normalizez matricea initiala scazand din ea media fiecarei linii astfel:

*calculez media fiecarei linii facand suma pe fiecarei linie si dupa impartind la numarul de coloane n

*scad din fiecarea element de pe linie i a matricei imaginii media liniei respective

- construiesc matricea Z

- calculez matricile U, S si V prin aplicarea SVD pe matricea Z

- construiesc matricea W din primele pcs coloane ale lui V

- caclulez matricea Y

- aproximez matricea initiala cu W * Y + miu

- castez matricea cu aproximatia la uint8

task3:

- castez matricea imaginii ‘photo’ la double

- calculez media fiecarei linii facand suma pe fiecarei linie si dupa impartind la numarul de coloane n

- normalizez matricea imaginii scazand din fiecarea element de pe linie i media liniei respective

- calculeaza matricea de covarianta

- calculeaza vectorii si valorile proprii ale matricei de covarianta

- pun valorile proprii intr-un vector pe care il sortez descrescator

- folosesc functia sort care intorce si un vector cu vechile pozitii ale elementelor din vectorul sortat pe care il folosesc pentru a ordona vectorii proprii

- pastreaza doar primii pcs vectori proprii

- creaza matricea Y schimband baza matricei imaginii

- calculeaza matricea new_image care este o aproximatie a matricii initiale

- castez matricea new_image la uint8

Task4

1. prepare_data:

- incarc datele din tabelul primit ca argument

- salveaza in matricea train_mat primele no_train_images linii din tabelul de imagini de antrenament

- salveaza in vectorul train_val primele no_train_images valori ale vectorului de etichete

2. visualise_image:

- citesc din matricea de antrenament linia cu numarul number

- transform linia citita intr-o matrice 28x28 pe care mai apoi o transpuspun

- castez matricea la unit8

3. magic_with_pca:

- cast train_mat la double => matricea X

- calculeaza media fiecarei coloane a matricei X

- scad media din matricea X

- calculez matricea de covarianta

- calculeaza vectorii si valorile proprii ale matricei de covarianta cu functia eig

- pun valorile proprii intr-un vector pe care il sorzez descrescator

- folosesc functia sort care intorce si un vector cu vechile pozitii ale elementelor din vectorul sortat pe care il folosesc pentru a ordona vectorii proprii

- pastrez doar primii pcs vectori proprii

- creez matricea Y schimband baza matricei initiale

- calculeaza matricea train care este o aproximatie a matricei initiale folosindu-ma de matricea cu primii pcs vectori proprii

4. prepare_photo:

- inversez pixelii imaginii primite

- trasnpun imaginea si dupa o transform intr-un vector linie

5. KNN:

- pentru fiecare linie(imagine din setul de antrenament) calculaleaza distanta Euclidiana dintre aceasta si imaginea de test

- ordoneaza crescator distantele si pastresza un vector cu vechile pozitii in cadrul setului de imagini(a cincea imagine, a noua imagine, ...)

- selectez labelurile pentru acele k imagini cu distanta cea mai mica si calculez predictia ca mediana labelurilor selectate

6. classifyImage:

- castez imaginea la double

- aplic functia magic_with_pca setului de date de antrenament

- scad din vectorul image media fiecarei coloane din matricea de antrenament initiala

- proiectez vectorul image pe spatiul componentelor principale prin inmultire cu matricea Vk

- calculeaza predictia cu metoda k nearest neighbour pentru k = 5 folosing proiectiile setului de date si a imaginii calculate anterior