from sklearn.datasets import fetch\_openml

from sklearn.model\_selection import train\_test\_split

from sklearn.svm import SVC

from sklearn.metrics import accuracy\_score

from sklearn.externals import joblib

# Adatok betöltése a MNIST adatkészletből

mnist = fetch\_openml('mnist\_784')

# A bemenetek (X) és a kimenetek (y) elkülönítése

X = mnist.data

y = mnist.target

# Az adatok normalizálása (0-255 skálából 0-1 skálára)

X = X / 255.0

# Az adatok felosztása tanítási és tesztadatra (80%-20% arányban)

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

# SVM modell inicializálása és betanítása

model = SVC(kernel='linear', C=10, gamma=0.01)

model.fit(X\_train, y\_train)

# A modell pontosságának kiszámítása a tesztadaton

y\_pred = model.predict(X\_test)

accuracy = accuracy\_score(y\_test, y\_pred)

print("Accuracy: {:.2f}%".format(accuracy \* 100))

# A betanított modell elmentése a memóriából

joblib.dump(model, "digits\_cls.pkl")

import cv2

from sklearn.externals import joblib

from skimage.feature import hog

# Betöltjük a betanított modellt

model = joblib.load("digits\_cls.pkl")

# Kézírásos számjegy felismerése

def recognize\_digit(img):

# Kép előkészítése a felismerésre

gray = cv2.cvtColor(img, cv2.COLOR\_BGR2GRAY)

ret, th = cv2.threshold(gray, 0, 255, cv2.THRESH\_BINARY\_INV+cv2.THRESH\_OTSU)

contours, hierarchy = cv2.findContours(th, cv2.RETR\_EXTERNAL, cv2.CHAIN\_APPROX\_SIMPLE)

# Minden kontúr köré írt téglalapba illesztjük az eredeti képen

rects = [cv2.boundingRect(ctr) for ctr in contours]

for rect in rects:

# Kivágjuk a képrészletet, amely a kontúr körül van

x, y, w, h = rect

if w > 8 and h > 8:

roi = gray[y:y+h, x:x+w]

roi = cv2.resize(roi, (28, 28), interpolation=cv2.INTER\_AREA)

# Az adatok átalakítása a modell által elvárt formátumba

roi\_hog\_fd = hog(roi, orientations=9, pixels\_per\_cell=(14, 14), cells\_per\_block=(1, 1), visualize=False)

roi\_hog\_fd = roi\_hog\_fd.reshape(1, -1)

# A modell predikciója

nbr = model.predict(roi\_hog\_fd)

cv2.putText(img, str(int(nbr[0])), (x, y), cv2.FONT\_HERSHEY\_DUPLEX, 1, (0, 255, 255), 2)

return img

# A kép beolvasása

img = cv2.imread("kep.jpg")

# Az előző függvényt használjuk a kép felismerésére

recognized\_img = recognize\_digit(img)

# Az eredmény megjelenítése

cv2.imshow("Recognized Image", recognized\_img)

cv2.waitKey(0)

cv2.destroyAllWindows()