KÉPFELISMERÉS GÉPI TANULÁSI MÓDSZEREK ALKALMAZÁSÁVAL

ÖNÁLLÓ LABORGYAKORLAT

KÉSZÍTETTE: KÖŐ JÓZSEF

TÉMAVEZETŐ: CZÚNI LÁSZLÓ

Áttekintés

- Célkitűzés
- Gépi tanulás áttekintése
- Fejlesztői eszközök
- Felhasznált adatforrások
- Osztályozó algoritmusok bemutatása
- Elért eredmények

Célkitűzés

Gépi tanulási eljárások megismerése, megvalósítása

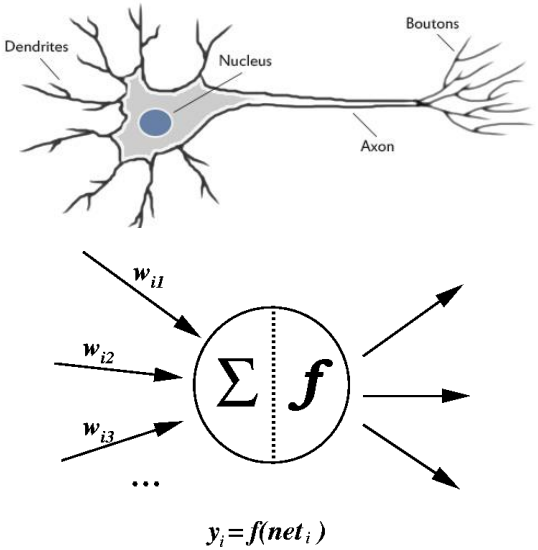
Kapcsolódó eszköztárral való ismerkedés

Tananyag feldolgozása – CS231n Course, Stanford University

Gépi tanulás

*„A gépi tanulás a számítástudomány azon ágazata amely a számítástechnikai eszközöket ruházza fel a **tanulás** képességével adatok alapján, **közvetlen programozás nélkül**.”*

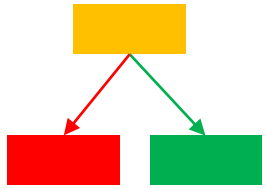
Biológiai analógia



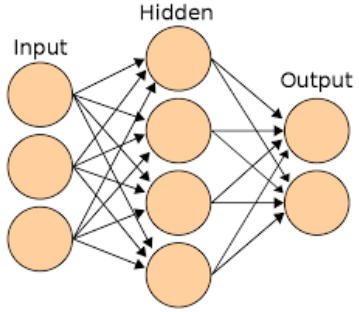
Alkamazási területek

Tanulási algoritmusok

Döntési fa



Mesterséges neurális háló



Megismert fejlesztői eszközök

Python 2.7

Anaconda

Spyder

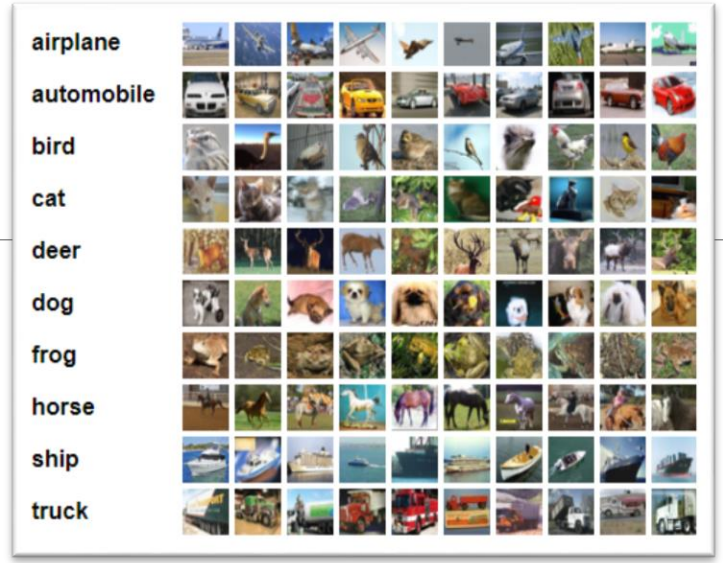
Numpy könyvtár


```
55  """
56  ----------------- SVM function definition -------------
57  -----------------  with integrated bias   -------------
58  """
59  #init variables
60  X = np.append(data[0],[1])
61  B = np.reshape(B,(1,K))
62  W = np.append(W,B.T,axis=1)
63
64  def SVM_Scores_bias(x,W):
65      #calculate scores
66      scores = W.dot(x)
67      return scores
68  s2=SVM_Scores_bias(X,W)
69  #print(s2)
```

Felhasznált adatforrások

CIFAR-10

- 60000 db 32x32 színes kép
- 10 osztály



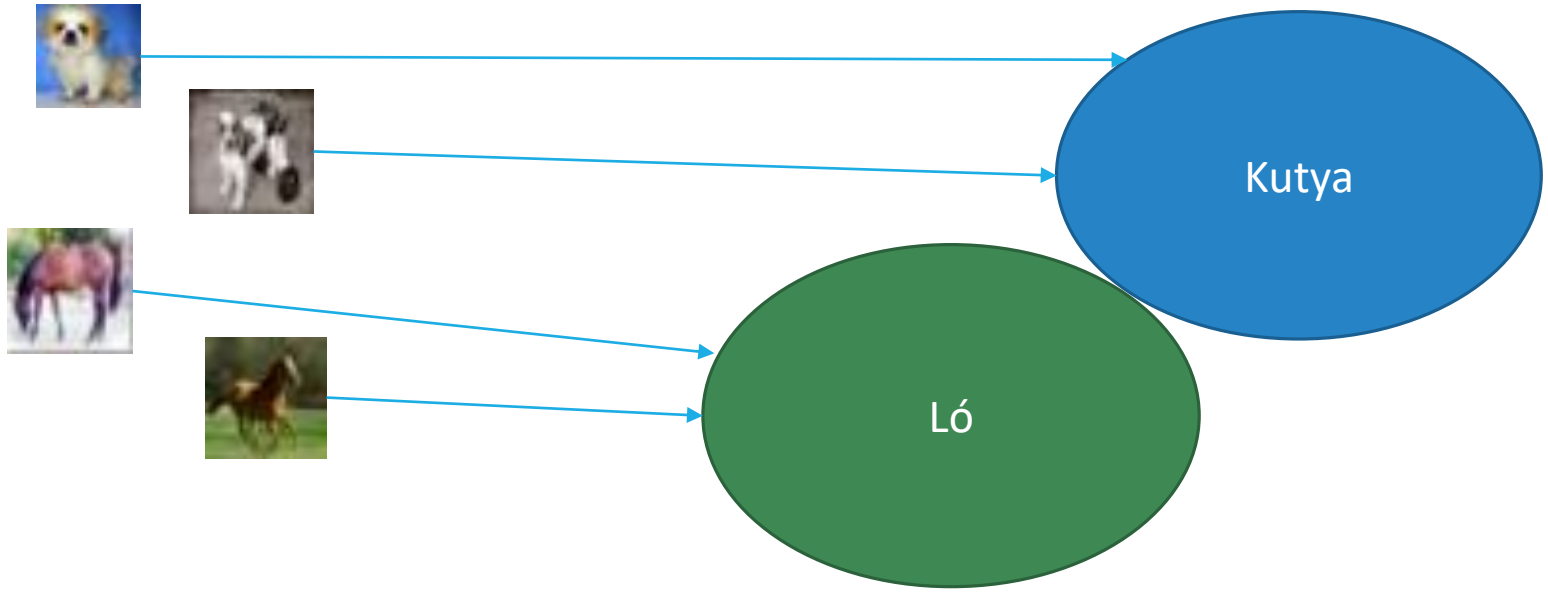
ImageNET

```
ıttp://www.nea.gov.sg/cms/pcd/car_image8.gif
ıttp://farm4.static.flickr.com/3048/2690941404_f5a75a4568.jpg
ıttp://farm3.static.flickr.com/2200/1562305641_7e37719fc0.jpg
ıttp://farm4.static.flickr.com/3096/2436911895_43acf498d0.jpg
ıttp://www.scienceclarified.com/everyday/images/scet_02_img0125.jpg
ıttp://www.vehicle-tracker.org.uk/images/vehicle-tracker-norfolk.jpg
ıttp://img.alibaba.com/photo/12035691/Ev_Electric_Vehicle_Electric_Car.jpg
ıttp://farm3.static.flickr.com/2420/2252055675_1ec3de9c46.jpg
ıttp://farm4.static.flickr.com/3223/2716629316_7997d2174f.jpg
ıttp://farm4.static.flickr.com/3146/2716595106_b48cab5f59.jpg
ıttp://farm1.static.flickr.com/29/90623098_7730e7e18e.jpg
ıttp://farm4.static.flickr.com/3216/2416297192_b4ff2a87d8.jpg
ıttp://farm4.static.flickr.com/3079/2665649099_1d6ec31382.jpg
```

```
21  #read base images
22  def unpickle(file):
23      import cPickle
24      with open(file, 'rb') as fo:
25          dict = cPickle.load(fo)
26      return dict
27
28  dictionary = unpickle('../info/cifar-10-batches-py/data_batch_1') #loading first batch
29  data = dictionary['data']
30  labels = dictionary['labels']
```

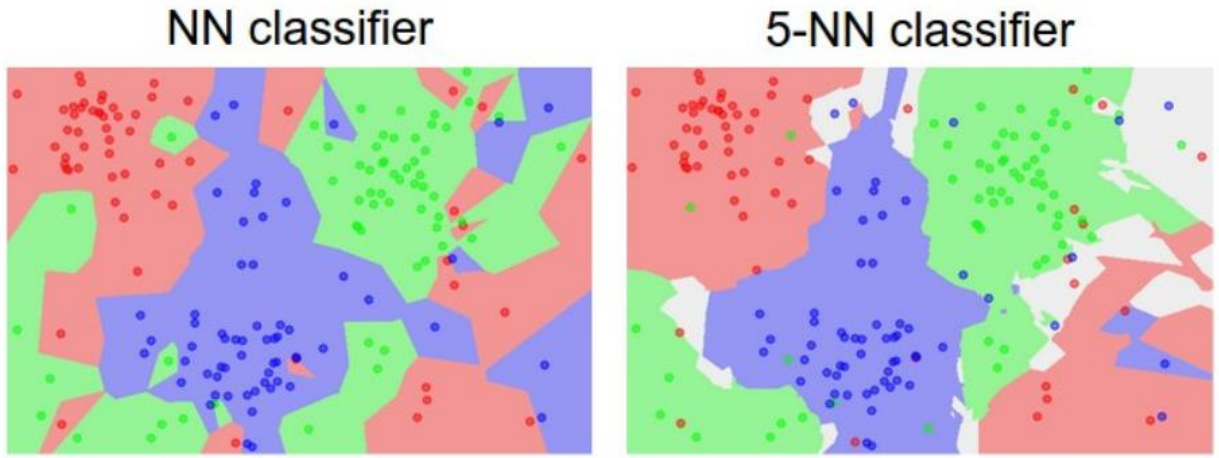
Példafeladat

Osztályozzuk a bemenetként kapott képeket a rajtuk szereplő dolgok szerint!



Osztályozó algoritmusok bemutatása K-legközelebbi szomszéd

- Paramétertérben tárolt pontok
- Új értékek a pontok alapján határozódnak meg




```
51    def train(self, X, y):
52      """ X is N x D where each row is an example. Y is 1-dimension of size N """
53      # the nearest neighbor classifier simply remembers all the training data
54      self.Xtr = X
55      self.ytr = y
```

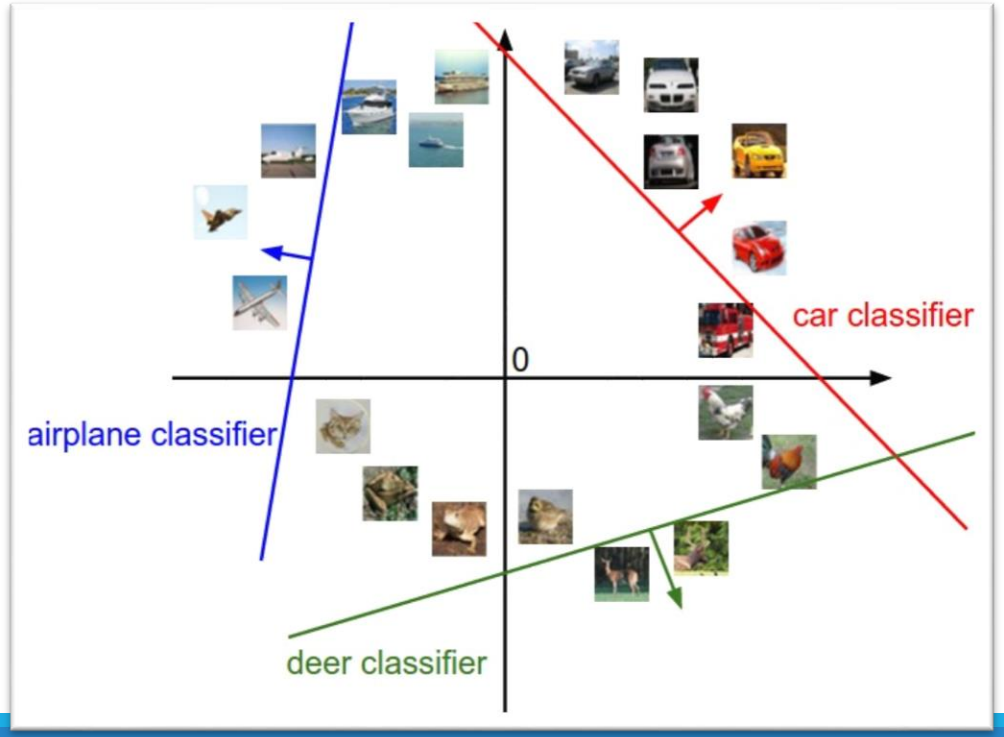
```
57    def predict(self, X):
58      """ X is N x D where each row is an example we wish to predict label for """
59      num_test = X.shape[0]
60      print X.shape[0]
61      # lets make sure that the output type matches the input type
62      Ypred = np.zeros(num_test)
63
64      # loop over all test rows
65      for i in xrange(num_test):
66        # find the nearest training image to the i'th test image
67        # using the L1 distance (sum of absolute value differences)
68        distances = np.sum(np.abs(self.Xtr - X[i]), axis = 1)
69        min_index = np.argmin(distances) # get the index with smallest distance
70        Ypred[i] = self.ytr[min_index] # predict the label of the nearest example
```

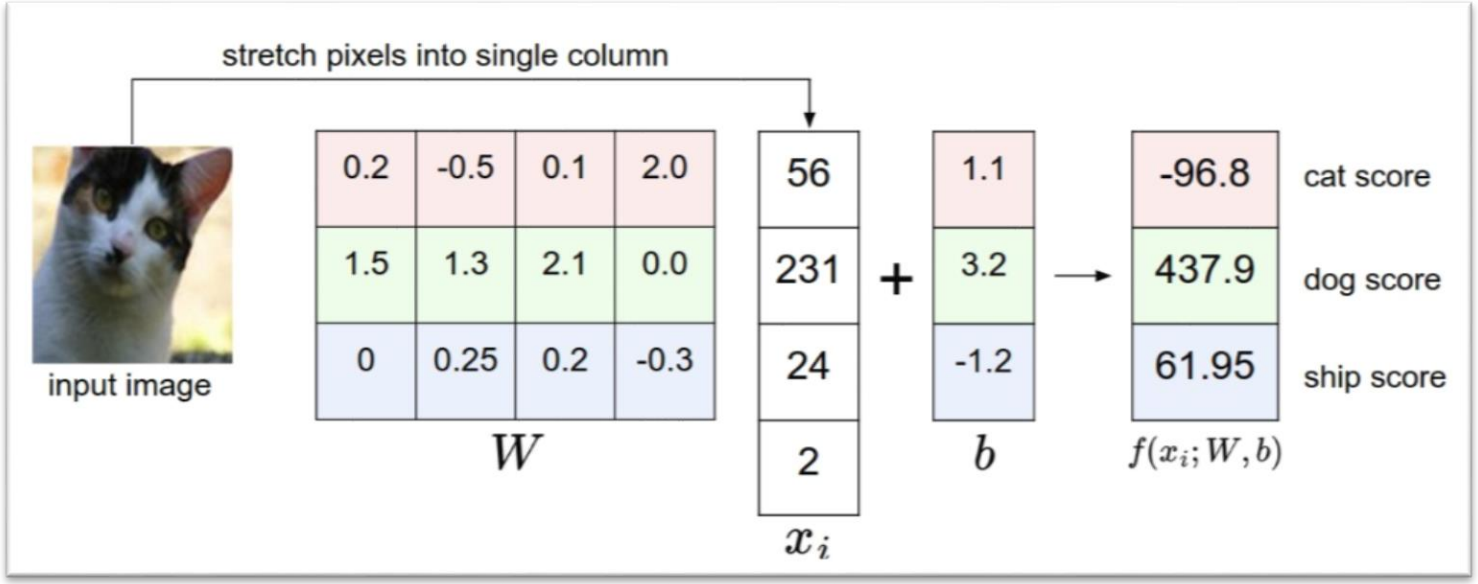
SVM – Support Vector Machine

Paramétertérben pontok

Hipersíkkal szeparálja a pontokat

$$f(x_i, W, b) = Wx_i + b$$





SVM - tanítás

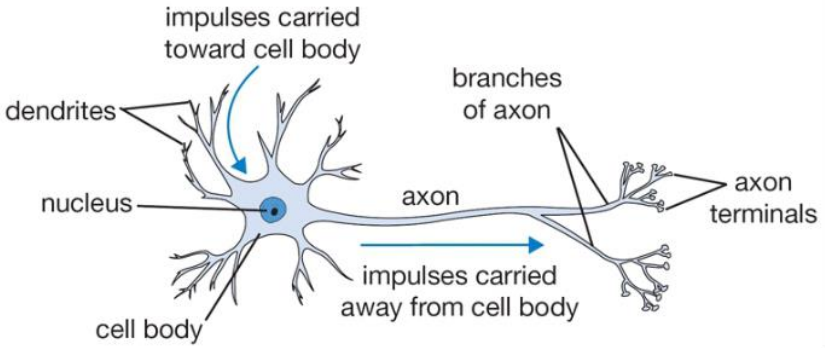
- Veszteségfüggvény kiszámítása
- Veszteségek minimalizálása

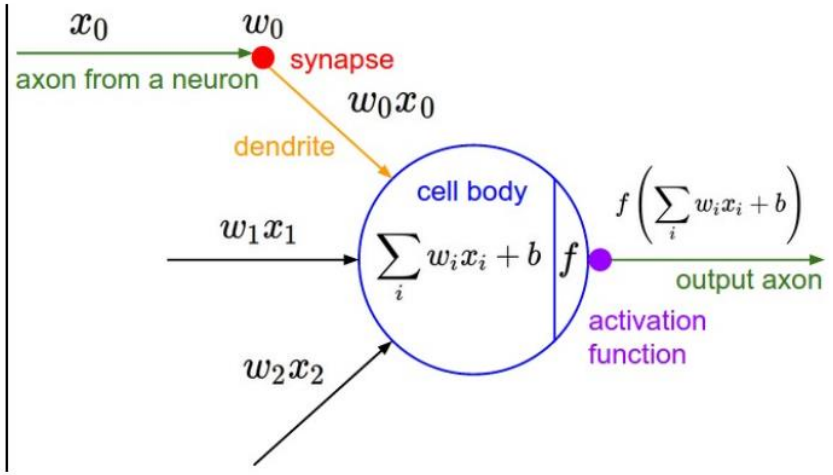
- Véletlen W értékek
- Gradiens módszer

$$L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + \Delta)$$

```
104  def L_i_vectorized(x, y, W):
105      """
106      A faster half-vectorized implementation. half-vectorized
107      refers to the fact that for a single example the implementation contains
108      no for loops, but there is still one loop over the examples (outside this function)
109      """
110      delta = 1.0
111      # compute the margins for all classes in one vector operation
112      margins = np.maximum(0, s2 - s2[y] + delta)
113      # on y-th position scores[y] - scores[y] canceled and gave delta. We want
114      # to ignore the y-th position and only consider margin on max wrong class
115      margins[y] = 0
116      loss_i = np.sum(margins)
117      return loss_i
118
119  loss = L_i_vectorized(X,labels[0],W)
120  #print (loss)
```

Neuron



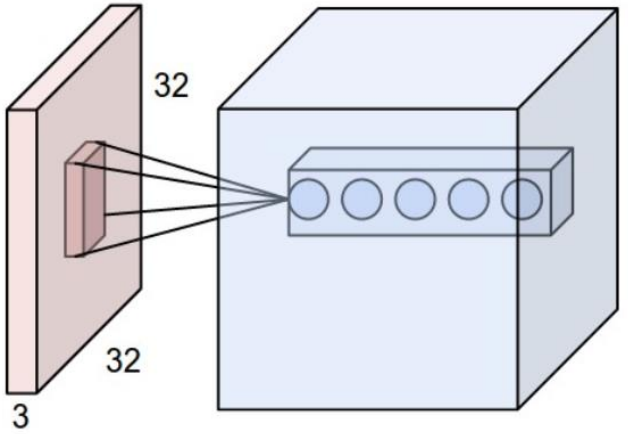



```python
class Neuron(object):
  # ...
  def forward(self, inputs):
    """ assume inputs and weights are 1-D numpy arrays and bias is a number """
    cell_body_sum = np.sum(inputs * self.weights) + self.bias
    firing_rate = 1.0 / (1.0 + math.exp(-cell_body_sum)) # sigmoid activation function
    return firing_rate
```

Neurális hálók

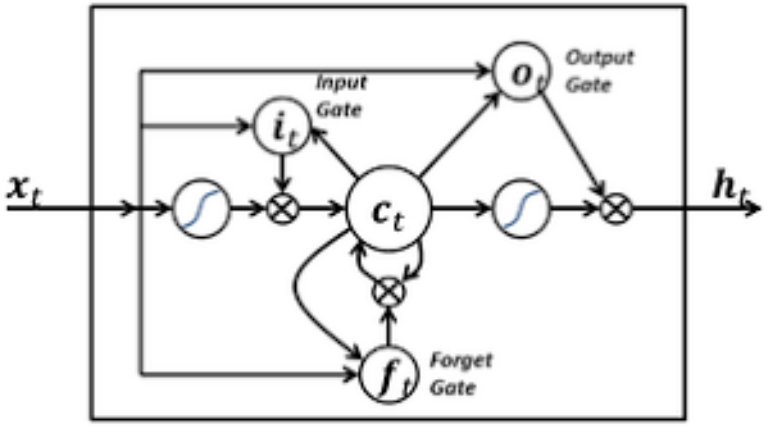
Konvolúciós hálók

- Képfeldolgozás
- Objektumfelismerés



LSTM hálók (Long-short term memory)

- Időbeli sorozat jóslata
- Ritmus tanulás



Elért eredmények

Gépi tanulási eljárások megismerése

Megvalósításuk

Kapcsolódó eszköztárral való ismerkedés

Tananyag feldolgozása

Köszönöm a figyelmet!

Kurzus link: **http://cs231n.stanford.edu/syllabus.html**

Projekt elérhetősége: **https://github.com/koojozsef/AI-project-for-university**

CIFAR-10 adatbázis: **https://www.cs.toronto.edu/~kriz/cifar.html**