Instantly share code, notes, and snippets.

anteq / cnn.py

Last active 6 days ago

Embed ▾ a8cc30b3f3ea.js"></script> Download ZIP

Condensed Nearest Neighbors task for ML course

cnn.py

```
# Antoni Grzanka
# implementation based on SciKit tutorial
# http://scikit-learn.org/stable/auto_examples/neighbors/plot_classification.html

import numpy as np
import matplotlib.pyplot as plt
from matplotlib.colors import ListedColormap
from sklearn import neighbors, datasets
from itertools import product
from sklearn.neighbors import DistanceMetric
from imblearn.under_sampling import CondensedNearestNeighbour

iris = datasets.load_iris()
X = iris.data[:, :2]
y = iris.target

cnn = CondensedNearestNeighbour()
X_cnn, y_cnn = cnn.fit_sample(X, y)

# Create color maps
cmap_light = ListedColormap(['#FFAAAA', '#AAFFAA', '#AAAAFF'])
cmap_bold = ListedColormap(['#FF0000', '#00FF00', '#0000FF'])

metrics = ['euclidean', 'mahalanobis']

n_neighbors = [1,3]

datasets = [{"X": X, "y": y, "cnn": False}, {"X": X_cnn, "y": y_cnn, "cnn": True}]

for metric, n, data in product(metrics, n_neighbors, datasets):

    X = data["X"]
    y = data["y"]

    if ( metric == 'mahalanobis' ):
        params = { "V": np.cov(X[:, 0], X[:, 1], rowvar=0) }
    else:
        params = None

    clf = neighbors.KNeighborsClassifier(n, weights='distance', metric=metric, metric_params=params)
    clf.fit(X, y)

    # Plot the decision boundary. For that, we will assign a color to each
    # point in the mesh [x_min, x_max]x[y_min, y_max].
    x_min, x_max = X[:, 0].min() - 1, X[:, 0].max() + 1
    y_min, y_max = X[:, 1].min() - 1, X[:, 1].max() + 1
    xx, yy = np.meshgrid(np.arange(x_min, x_max, .02), np.arange(y_min, y_max, .02))
    Z = clf.predict(np.c_[xx.ravel(), yy.ravel()])

    # Put the result into a color plot
    Z = Z.reshape(xx.shape)
    plt.figure()
    plt.pcolormesh(xx, yy, Z, cmap=cmap_light)

    plt.scatter(X[:, 0], X[:, 1], c=y, cmap=cmap_bold)
    plt.xlim(xx.min(), xx.max())
    plt.ylim(yy.min(), yy.max())
    plt.title("k = %i, metric = '%s', cnn = '%s'" % (n, metric, data["cnn"]))

plt.show()
```