Домашнее задание №8

Deadline: 20.05.2017 23:59:59

ФИВТ, АПТ, Курс по машинному обучению, Весна 2017, Модуль Unspervised Learning,

Alexey Romanenko, alexromsput@gmail.com

Organization Info

Дополнительный материал для выполнения дз:

- Воронцов К. В. Математические методы обучения по прецедентам. 2012. http://www.machinelearning.ru/wiki/images/6/6d/Voron-ML-1.pdf (http://www.machinelearning.ru/wiki/images/6/6d/Voron-ML-1.pdf) (разделы 5.2 и 7.1)
- Hastie T., Tibshirani R., Friedman J. The Elements of Statistical Learning. Springer: Data Mining, Inference, and Prediction. — 2nd ed. — Springer-Verlag. 2009. — 746 p.http://statweb.stanford.edu/~tibs/ElemStatLearn/printings/ESLII_print10.pdf (http://statweb.stanford.edu/~tibs/ElemStatLearn/printings/ESLII_print10.pdf) (глава 14)

Оформление дз:

- Присылайте выполненное задание на почту `ml.course.mipt@gmail.com`
- Укажите тему письма в следующем формате `ML2017_fall` <номер_группы> <фамилия>, к примеру -- `ML2017_fall 496 ivanov`
- Выполненное дз сохраните в файл <фамилия>_<группа>_task<номер задания>.ipnb, к примеру -- `ML2017_496_task1.ipnb`

Вопросы:

- Присылайте вопросы на почту `ml.course.mipt@gmail.com`
- Укажите тему письма в следующем формате `ML2016_fall Question` <Содержание вопроса>

- **PS1**: Используются автоматические фильтры, и просто не найдем ваше дз, если вы не аккуратно его подпишите.
- **PS2**: Дедлайн жесткий, в том числе помтоу что это ДЗ последнее в курсе.

Контрольные вопросы (0 % - для самоконтроля)

Ответте на вопросы своими словами (загуглянный материал надо пересказать), ответ

Ответе на вопросы своими словами (заглуленный материал надо пересказать), ответ обоснуйте (напишите и ОБЪЯСНИТЕ формулки если потребуется), если не выходит, то вернитесь к лекции дополнительным материалам:

Вопрос 1: В чём заключается проблема мультиколлинеарности? При минимизации ошибки Q с МНК матрица $(F^TF)^{-1}$ может иметь очень большие значения, и оценка α становится неустойчивой.

Вопрос 2: Какие проблемы при обучении алгоритмов возникают из-за большой размерности пространства признаков?

Основные проблемы: 1) Мультиколлинеарность 2) Неинтерпретируемость 3) Проблема размерности

Вопрос 3: В чем суть проклятия размерности?

Проблема, связанная с экспоненциальным возрастанием количества данных из-за увеличения размерности пространства.

Вопрос 4: Какая связь между решением задачи PCA и SVD-разложение матрицы регрессии?

Решение задачи PCA находится при помощи сингулярного разложения (SVD)

Вопрос 5: Почему в tSNE расстояние между парамми объектов измеряется "по-стьюденту" и как это помогает решить проблему "скрученности" (crowding problem)?

Вопрос 6: На какой идее базируются алгоритмы агломеративной кластеризации? Напишите формулу Ланса-Вильма

Агломеративные методы кластеризации основываются на принципе последовательного объединения данных в компоненты. Изначально все элементы представляют собой отдельные компоненты, затем постепенно ближайшие компоненты начинают объединяться. Например, формула Ланса-Вильма — универсальный метод вычисления расстояния между компонентами:

$$R(U \cup V, S) = \alpha_U R(U, S) + \alpha_V R(V, S) + \beta R(U, V) + \gamma |R(U, S) - R(V, S)|$$

Вопрос 7: Какие два шага выделяют в алгоритме кластеризации k-means?

1) отнесение объектов к кластерам 2) пересчёт центров кластеров

Вопрос 8: В чём отличия (основные упрощения) k-means от EM-алгоритма кластеризации?

В алгоритме k-means кластеры - центроиды, то есть, по сути, предполагается, что кластеры - это некоторые сферические объекты, в EM-алгоритме, присутствует расширенное описание кластеров, объект имеет вероятностное распределение по кластерам, формы кластеров могуть быть любыми, они могут и перекрываться, что полностью сломало бы k-means.

Вопрос 9 Какой принцип работы графовых алгоритмов кластеризации?

Графовые алгоритмы кластеризации представляют данные в виде графа, где на рёбрах записаны расстояния между вершинами.

Вопрос 10 В чем некорректность постановки задачи кластеризации?

PS: Если проверяющий не понял ответ на большинство вопросов, то будет пичалька. Пишите так, чтобы можно было разобраться.

Вопросы по теории (30%)

Задача 1 Ответьте на вопросы:

1) Как можно не прибегая к визуализации понять, что кластерная структура у данного облака точек отсутствует? 2) Какие из алгоритмов кластеризации могут выделять кластеры с ленточной структурой? 3) Какие алгоритмы кластеризации чувствительны к шуму и перемычкам? 4) Каким образом приближают «центр кластера» в нелинейных пространствах? 5) Каким образом можно определять число кластеров?

1. Применить к-н алгоритм кластеризации несколько раз, например, EM. Если алгоритм каждый раз будет сходится к существенно разным результатам, то скорее всего кластерная структура отсутствует.
2. Алгоритм выделения связных компонент, Кратчайший незамкнутый путь, ФОРЕЛ, DBSCAN.
3. Алгоритм выделения связных компонент, Кратчайший незамкнутый путь, Ланса-Уильямса
4. Пусть есть кластер K и метрика $\rho(x, x')$, тогда центр можно определить так:
$$x_{center} = arg \min_{x \in K} \sum_{x' \in K} \rho(x, x')$$
5. Наличие кластерной структуры можно проверить с помощью статистики Хопкинса, количество кластеров с помощью коэффициента силуэта.

Задача 2 Даны пять точек на числовой оси $X = (1; 5; 7; 8; 8)$, число кластеров равно 2. Рассчитайте ответ алгоритма K-means (финальные центры кластеров), если начальные центры кластеров c1 = 1, c2 = 10.

После первого шага: $y_1 = \{1, 5\}, y_2 = \{7, 8, 8\}, \mu_1 = 3, \mu_2 = 7\frac{1}{3}$

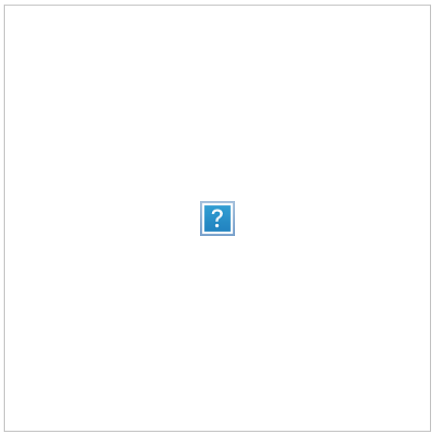
После второго шага: $y_1 = \{1, 5\}, y_2 = \{7, 8, 8\}, \mu_1 = 3, \mu_2 = 7\frac{1}{3}$

Кластеризация не изменилась, поэтому алгоритм на этом и остановится.

Задача 3 Докажите, что the k-means всегда сходится.

Пусть алгоритм не сходится, то есть алгоритм все время переходит от одной кластеризации к другой с меньшим cost. Возможных k^n кластеризаций < inf, а, значит, k-means когда-то войдет в цикл, что невозможно, так как cost все время уменьшается, противоречие => алгоритм сойдется.

Задача 4 Для сжатия размерности пространства алгоритм PCA применяется датасету с количеством признаков $D = 100$. Наблюдается следующий спектр собственных значений матрицы объектов-признаков.



Ответье на вопросы

- 1) Высокая ли эффективная размерность пространства признаков (intrinsic dimensionality) (насколько она близка к 100)? Высокая. почти нигде нет переломного момента, после которого спект резко падает.
- 2) Можно ли перевести датасет с помощью PCA в пространство меньшей размерности с минимальными потерями точности? Если да, то чему примерно будет равна размернось Нельзя

Практическое задание 1 (30%)

Реализуйте PCA

In [2]:
```
import numpy as np
import pylab as plt
import sklearn as sk
import matplotlib.pyplot as plt
%matplotlib inline
'''
Performs the Principal Coponent analysis of the Matrix F
Matrix must be n * l dimensions
where n is # features
l is # samples
'''

def PCA(F, varRetained = 0.95, show = False):
    # Input
    #     F - initaial matrix
    # Compute Covariance Matrix Sigma
    # Input
    (n, l) = F.shape
    Sigma = 1.0 / l * np.dot(F, np.transpose(F))
    # Compute eigenvectors and eigenvalues of Sigma by SVD
    # U, V - matrix, d - array: Sigma = U * np.diag(d) * V
    U, d, V = np.linalg.svd(Sigma)

    # compute the value m: number of minumum features that retains the
    dTot = np.sum(d)
    var_i = np.array([np.sum(d[: i + 1]) / \
                dTot * 100.0 for i in range(n)])
    for i,v in enumerate(var_i):
        if v > varRetained * 100:
            break
    m = i
    print '{:,.2f} variance retained in {} dimensions'.format( varReta
    if show:
        plt.plot(var_i)
        plt.xlabel('Number of Features')
        plt.ylabel(' Percentage Variance retained')
        plt.title('PCA $\% \sigma^2 $ vs # features')
        plt.show()

    # compute the reduced dimensional features by projection
    U_reduced = U[:, :m]
    G = (np.dot(U, np.diag(d)))[:, :m]
    return G, U_reduced
```

In [2]:
```
# Примените алгоритм к данным MNIST
from sklearn.model_selection import train_test_split

from sklearn.datasets import load_digits
X, y = load_digits(return_X_y=True)
X_train, X_test, Y_train, Y_test = train_test_split(X, y, train_size=0
```

In [3]:
```
##################################################################
# PCA of training set
print 'Performing PCA - Principal COmponent Analysis'

Z, U_reduced = PCA(X.T, varRetained = 0.95, show = False)
```

```
Performing PCA - Principal COmponent Analysis
0.95 variance retained in 15 dimensions
```

In [4]:
```
print Z
print U_reduced
```

```
[[  0.00000000e+00   0.00000000e+00   0.00000000e+00   0.00000000e
+00
    0.00000000e+00   0.00000000e+00   0.00000000e+00   0.00000000e
+00
    0.00000000e+00   0.00000000e+00   0.00000000e+00   0.00000000e
+00
    0.00000000e+00   0.00000000e+00   0.00000000e+00]
 [ -1.54488948e+01  -3.10607025e+00   1.61146465e+00  -2.60319317e
+00
    2.01100765e+00  -9.87497834e-01   6.26867673e-01  -4.31849803e
-01
    8.24296592e-01  -8.03695500e-01   4.06519978e-01  -2.44323977e
-01
   -1.37001215e+00   4.73831836e-01  -3.02828618e-01]
 [ -2.69518610e+02  -4.01097776e+01   7.32497095e+00  -1.74114581e
+01
    1.83490771e+01  -6.05705316e+00   3.52884908e+00  -2.79240905e
+00
    9.25282128e+00  -3.31234727e+00  -1.29583417e+00  -3.39274459e
+00
```

Практическое задание 2 (40%)

Изучение алгоритмов кластеризации на разных выборках

Кластеризация цифр с помощью dbscan

На данных из sklearn.datasets.load_digits примените алгоритмы кластеризации (знания о метках классов при кластеризации использовать нельзя):

- dbscan (http://scikit-learn.org/stable/modules/generated/sklearn.cluster.DBSCAN.html#sklearn.cluster.DBSCAN) запускайте при различных параметрах eps и minsamples, для всех экспериментов

можете выбрать одну метрику (вспомните семинар про метрические алгоритмы);
- Используя метки классов цифр, оцените качество различных кластеризаций при помощи Adjusted Mutual Information и Adjusted Rand Index.
- визуалируйте изображения тех цифр, которые соответствуют core_points;
- визуалируйте изображения тех цифр, которые соответствуют выбросам;
- сделайте выводы и применимости алгоритмов.

Уменьшение палитры изображения

- для картинки (https://thumbs.dreamstime.com/x/two-lorikeet-birds-2293918.jpg) нужно уменьшить число цветов в палитре; для этого нужно выделить кластеры в пространстве RGB, объекты соответствуют пикселам изображения; после выделения кластеров, все пикселы, отнесенные в один кластер, заполняются одним цветом; этот цвет может быть центроидом соответствующего кластера, медианным цветом по кластеру.
- Попробуйте различные алгоритмы кластеризации:

    -- KMeans
    -- MeanShift
    -- AgglomerativeClustering

  Рассмотрите число кластеров K = 2, 3, 10, 20
- Для различных кластеризаций оцените и сравните потери от уменьшения цветов при помощи метрики SSIM (http://scikit-image.org/docs/dev/api/skimage.measure.html). Какой способ оказался лучшим?

In [5]:
```
mnist = sk.datasets.load_digits()
```

In [8]:
```
from sklearn import cluster
```

In [7]:
```
digits = np.reshape(mnist.images, (mnist.images.shape[0], -1))
dbscan = cluster.DBSCAN(eps=21, min_samples=3, metric='euclidean', alg
labels = dbscan.fit_predict(digits)
```

In [10]:
```
sk.metrics.adjusted_mutual_info_score(labels, mnist.target)
```

Out[10]:
```
0.76524515153558503
```

In [11]:
```
sk.metrics.adjusted_rand_score(labels, mnist.target)
```

Out[11]:
```
0.59463144328200968
```

Попробуем найти наилучшие параметры

In [34]:
```
b_eps_i = 0
b_min_samples_i = 0
b_eps_r = 0
b_min_samples_r = 0
rand = 0
info = 0
for eps in np.linspace(3.5, 30., 100):
    for min_samples in np.arange(3, 20):
        dbscan = cluster.DBSCAN(eps=eps, min_samples=min_samples,
                                metric='euclidean', algorithm='auto',
        labels = dbscan.fit_predict(digits)
        i = sk.metrics.adjusted_mutual_info_score(labels, mnist.target
        r = sk.metrics.adjusted_rand_score(labels, mnist.target)
        if i > info:
            b_eps_i = eps
            b_min_samples_i = min_samples
            info = i
        if r > rand:
            b_eps_r = eps
            b_min_samples_r = min_samples
            rand = r
```

In [35]:
```
print(b_eps_i)
print(b_min_samples_i)
print(b_eps_r)
print(b_min_samples_r)
print(rand)
print(info)
```

```
21.1666666667
4
21.1666666667
4
0.673685734897
0.779628108702
```

In [36]:
```
dbscan = cluster.DBSCAN(eps=22, min_samples=4,
                                metric='euclidean', algorithm='auto',
labels = dbscan.fit_predict(digits)
dbscan.components_.shape
```

In [37]:
```
labels = dbscan.fit_predict(digits)
```

In [38]:
```
labels[:100]
```

Out[38]:
```
array([ 0,  1, -1,  2,  4, -1,  3,  6, -1, -1,  0,  1,  7,  2,  4,
5,  3,
        6,  1, -1,  0,  1,  7,  2,  4,  5,  3, -1, -1,  8,  0, -1,
5, -1,
        3, -1,  0, -1, -1,  8,  1,  4,  1,  6,  6,  2, -1,  1,  0,
0,  9,
        9,  6, -1, -1,  0,  1, -1,  3,  2,  2,  6,  2,  2,  4,  3,
3,  3,
        4, -1,  1,  5,  0,  8,  5, -1,  1, -1,  0,  0,  1,  6,  3,
2,  7,
        1, -1, -1,  3,  2,  1,  2,  8,  1,  6,  1, -1,  4,  2,  1])
```

In [39]:
```
dbscan.core_sample_indices_[:100]
```

Out[39]:
```
array([  0,   1,   3,   6,  10,  11,  13,  14,  15,  16,  17,  20,
21,
        22,  23,  24,  25,  26,  30,  32,  34,  36,  39,  40,  41,
42,
        44,  45,  47,  48,  49,  52,  55,  56,  58,  59,  60,  61,
62,
        63,  64,  65,  66,  67,  70,  71,  72,  73,  76,  78,  79,
80,
        81,  82,  83,  84,  85,  88,  89,  90,  91,  94,  97,  98,
99,
       100, 101, 102, 107, 108, 109, 111, 112, 114, 115, 117, 119, 1
24,
       126, 128, 130, 131, 132, 135, 136, 137, 139, 140, 141, 142, 1
43,
       144, 145, 146, 147, 149, 150, 151, 153, 154])
```

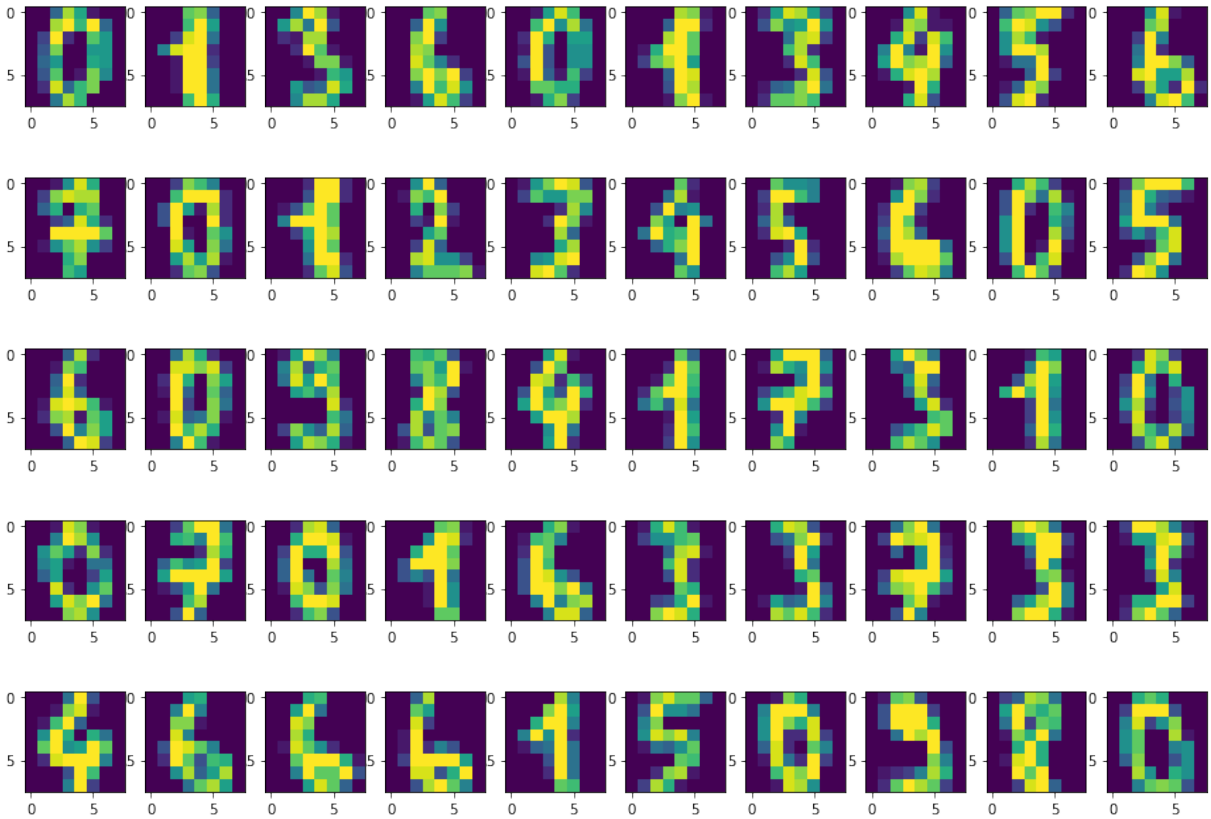
In [40]:
```
dbscan.components_.shape
```

Out[40]:
```
(1303, 64)
```

Это очень много! Фактически кластеризация не удалась. Посмотрим на центры.

In [41]:
```
def draw(images):
    plt.figure(figsize=(15, 10))
#     print(len(images))
    for i, image in enumerate(images):
        plt.subplot(i // 10 + 1, 10, i+1)
        plt.imshow(image)
#         plt.axis('off')
```

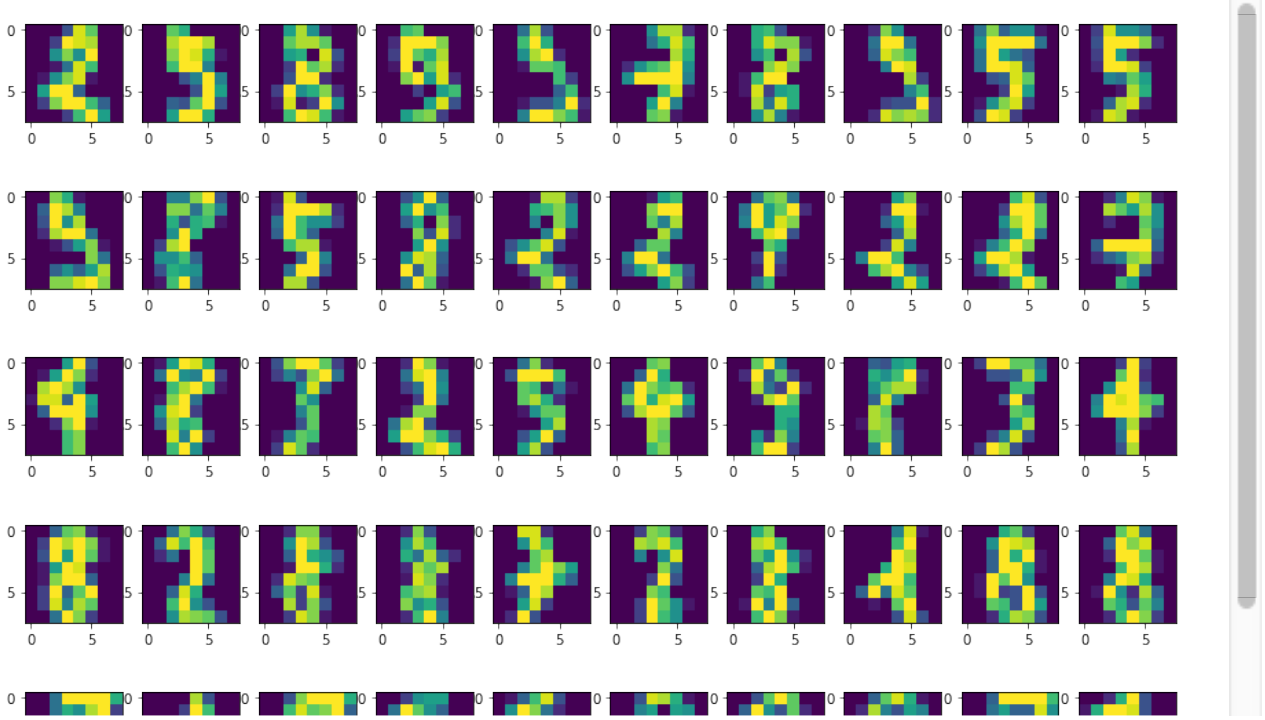
In [42]:
```
for i in range(5):
    draw(mnist.images[dbscan.core_sample_indices_[i * 10:(i + 1) * 10]
```



Вон там две почти одинаковые тройки подряд идут ;(

А это выбросы

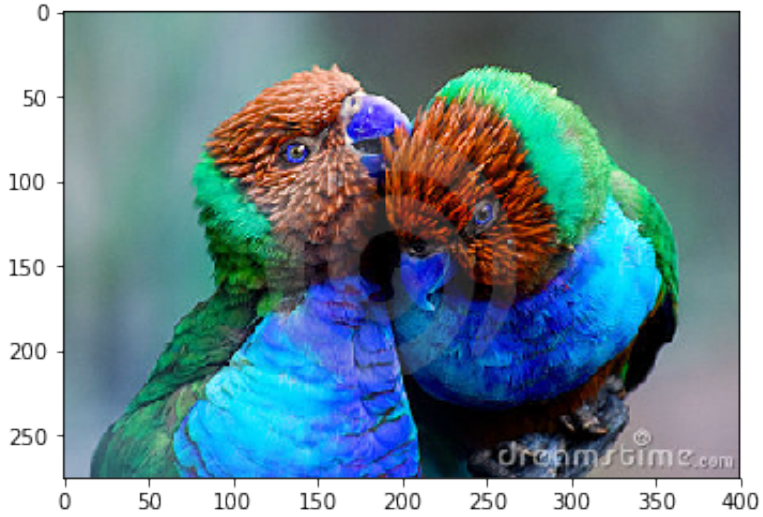
In [43]:
```
for i in range(5):
    draw(mnist.images[labels == -1][i * 10:(i + 1) * 10])
```



3.

In [36]:
```
from skimage import io
import cv2
img = cv2.imread('img.jpg')
plt.imshow(img)
img.shape
```

Out[36]: (275, 400, 3)



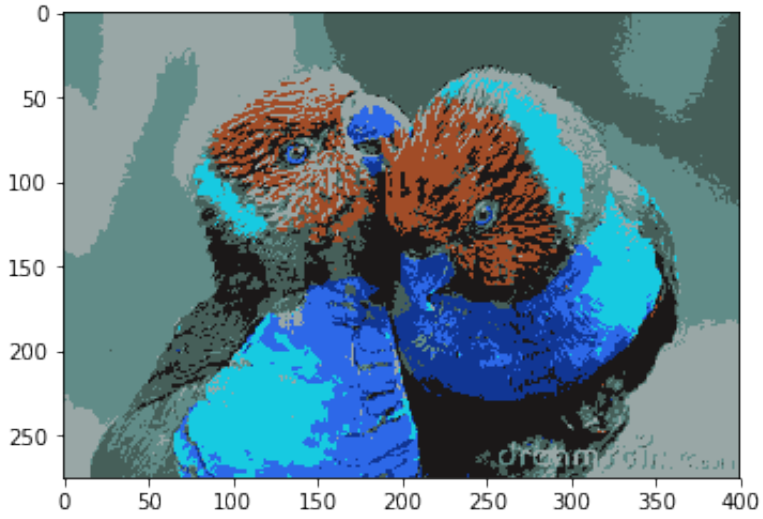
In [38]:
```
kmeans = cluster.KMeans()
```

In [47]:
```
from skimage import measure
import scipy.misc
```

In [51]:
```
img = np.reshape(img, (-1, 3))
labels = kmeans.fit_predict(img)
values = kmeans.cluster_centers_
img_processed = list(map(lambda (i, v): values[labels[i]], enumerate(i
img_processed = np.reshape(img_processed, (275, 400, 3))
img_processed = img_processed.astype(img.dtype)
img = np.reshape(img, (275, 400, 3))
print(measure.compare_ssim(img_processed, img, multichannel=True))
plt.imshow(img_processed)
```

```
0.655606723764
```

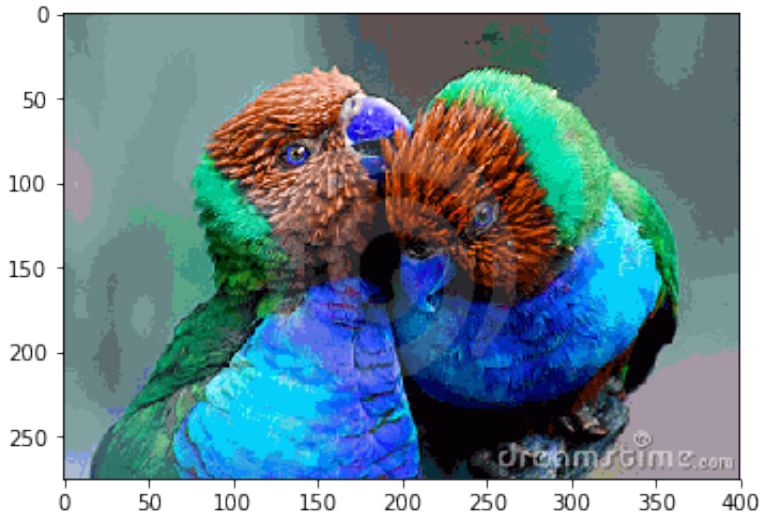
Out[51]: <matplotlib.image.AxesImage at 0x7ff947dbfa90>



In [53]:
```
msh = cluster.MeanShift(bandwidth=15)
img = np.reshape(img, (-1, 3))
labels = msh.fit_predict(img)
values = msh.cluster_centers_
img_processed = list(map(lambda (i, v): values[labels[i]], enumerate(i
img_processed = np.reshape(img_processed, (275, 400, 3))
img_processed = img_processed.astype(img.dtype)
img = np.reshape(img, (275, 400, 3))
print(measure.compare_ssim(img_processed, img, multichannel=True))
plt.imshow(img_processed)
```

```
0.852643312418
```

Out[53]:
```
<matplotlib.image.AxesImage at 0x7ff8eebcf790>
```



In [54]:
```
msh.cluster_centers_.shape
```

Out[54]:
```
(398, 3)
```

In [84]:
```
from sklearn import cluster
from sklearn.neighbors import kneighbors_graph
img = np.reshape(img, (-1, 3))
knn_graph = kneighbors_graph(img, 30, include_self=False)
agg = sk.cluster.AgglomerativeClustering(linkage='ward',
                                         connectivity=knn_graph,
                                         n_clusters=8)
labels = agg.fit_predict(img)
clusters = [img[labels == i] for i in range(agg.n_components_)]
values = []
print(clusters[0][:][2])
for i in range(agg.n_components_):
    values.append(np.mean(clusters[i], axis=0))
print(values)
img_processed = list(map(lambda (i, v): values[labels[i]], enumerate(i
img_processed = np.reshape(img_processed, (275, 400, 3))
img_processed = img_processed.astype(img.dtype)
img = np.reshape(img, (275, 400, 3))
print(measure.compare_ssim(img_processed, img, multichannel=True))
```

```
plt.imshow(img_processed)
```

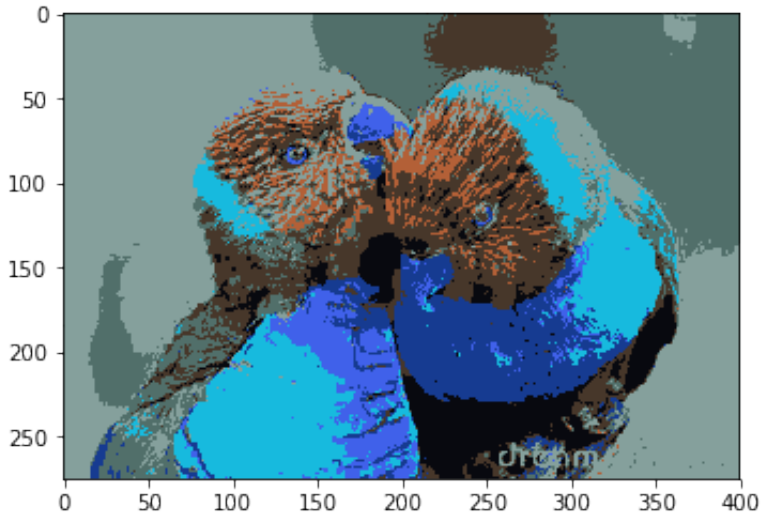
```
/usr/local/lib/python2.7/dist-packages/sklearn/cluster/hierarchical.
py:193: UserWarning: the number of connected components of the conne
ctivity matrix is 11 > 1. Completing it to avoid stopping the tree e
arly.
  connectivity, n_components = _fix_connectivity(X, connectivity)
```

```
[113 147 137]
[array([ 133.90430058,  160.63140344,  156.82967286]), array([  23.5
3779275,  186.35717083,  222.78499256]), array([ 71.40072202,  55.44
958742,  42.49207065]), array([  81.45904437,  111.90964613,  106.78
29621 ]), array([  22.99145779,   59.37495785,  145.74710577]), arra
y([ 179.5278335 ,   96.62462387,   55.67728185]), array([  64.041315
35,   97.64333895,  234.03920742]), array([  8.75362319,   9.1243250
9,  15.56393862]), array([ nan,  nan,  nan]), array([ nan,  nan,  na
n]), array([ nan,  nan,  nan])]
```

```
/usr/local/lib/python2.7/dist-packages/numpy/core/fromnumeric.py:288
9: RuntimeWarning: Mean of empty slice.
  out=out, **kwargs)
/usr/local/lib/python2.7/dist-packages/numpy/core/_methods.py:73: Ru
ntimeWarning: invalid value encountered in true_divide
  ret, rcount, out=ret, casting='unsafe', subok=False)
```

```
0.646406433364
```

Out[84]: <matplotlib.image.AxesImage at 0x7ff8ef06de10>



С кластеризацией на 8 цветов Kmeans и AgglomerativeClustering сработали почти одинаково по качеству, но на самом деле по-разному (видно визуально). MeanShift сработал лучше, но просто потому, что он кластеризовал на большее количество цветов&