In [1]:

```
import pandas

close_prices = pandas.read_csv(filepath_or_buffer='/Volumes/fast64/My/MachineLearnin
djia_index = pandas.read_csv(filepath_or_buffer='/Volumes/fast64/My/MachineLearning/

datas = close_prices['date']
close_prices = close_prices.drop('date', 1)
```

In [44]:

```
print close_prices([0])
print close_prices.info()
```

```
---------------------------------------------------------------------------
-----
TypeError                                 Traceback (most recent call
last)
<ipython-input-44-5eae7b00052f> in <module>()
----> 1 print close_prices([0])
      2 print close_prices.info()

TypeError: 'DataFrame' object is not callable
```

In [40]:

```
from sklearn.decomposition import PCA

pca = PCA(n_components=3)
pca.fit(close_prices)
```

Out[40]:

```
PCA(copy=True, n_components=3, whiten=False)
```

In [41]:

```
print sum(pca.explained_variance_ratio_)
print(pca.explained_variance_ratio_)
print(pca.components_)
```

```
0.898993755584
[ 0.73897118  0.11007169  0.04995088]
[[  1.61383840e-02   1.20644923e-01  -5.16612711e-02   5.04842369e-02
   -1.25859933e-01   1.14089567e-01   2.33906290e-01  -6.20513749e-03
    2.51227032e-01   2.88996029e-01  -2.64998795e-01   9.31320168e-02
    9.13948403e-02   4.69879340e-02   2.90549417e-02  -2.61068828e-02
    3.29615584e-01   7.13897133e-02   7.62295699e-02   2.11888868e-01
    2.30922941e-02   7.77316954e-02  -7.20594590e-03   1.89479745e-01
    3.21564017e-01   5.36834873e-02   5.79683946e-01   1.09122230e-04
    8.71614334e-02  -4.29421420e-02]
 [  2.33025791e-01  -1.38102207e-01   5.64736227e-01   2.94213771e-02
    2.97156167e-01   7.00723638e-02   2.41633004e-01   2.26028350e-02
    1.05527180e-01  -3.16032026e-02   3.56926513e-01   1.51980885e-01
    2.76101639e-01   6.73887479e-02   5.03438625e-02   5.87043804e-02
    2.17410195e-01   1.90769246e-01   1.66275222e-01  -2.25387182e-02
   -4.07006441e-02   1.86523497e-02   3.55593965e-02   7.67098774e-02
    5.41168982e-02   1.82997954e-03  -1.06387814e-01   3.21923154e-02
   -5.03033207e-02   2.57983104e-01]
 [  1.05902102e-01   4.73844545e-01   5.37487494e-02  -7.20638781e-03
    4.84014835e-02   6.66524068e-02   4.53663481e-02   3.17864061e-02
   -3.48308747e-01  -1.42316297e-01   2.06736652e-01  -1.59192377e-01
   -1.16708078e-01  -1.30026185e-02  -4.34870459e-02   1.87096064e-01
    8.54676853e-02   1.03314831e-02  -1.51788920e-01  -2.58569260e-01
    7.07110772e-02  -1.35307492e-01   1.10443628e-03  -9.18773088e-02
   -7.20122232e-02   4.33767778e-01   3.75349831e-01  -2.22521778e-02
    1.29157960e-02   1.64834697e-01]]
```

In [4]:

```
# Сколько компонент достаточно, чтобы описать 90% дисперсии?
pca2 = PCA(n_components=0.9)
pca2.fit(close_prices)

print(pca2.explained_variance_ratio_)
print(pca2.components_)
# answer 1 = 4
```

```
[ 0.73897118  0.11007169  0.04995088  0.0287492 ]
[[  1.61383840e-02   1.20644923e-01  -5.16612711e-02   5.04842369e-02
   -1.25859933e-01   1.14089567e-01   2.33906290e-01  -6.20513749e-03
    2.51227032e-01   2.88996029e-01  -2.64998795e-01   9.31320168e-02
    9.13948403e-02   4.69879340e-02   2.90549417e-02  -2.61068828e-02
    3.29615584e-01   7.13897133e-02   7.62295699e-02   2.11888868e-01
    2.30922941e-02   7.77316954e-02  -7.20594590e-03   1.89479745e-01
    3.21564017e-01   5.36834873e-02   5.79683946e-01   1.09122230e-04
    8.71614334e-02  -4.29421420e-02]
 [  2.33025791e-01  -1.38102207e-01   5.64736227e-01   2.94213771e-02
    2.97156167e-01   7.00723638e-02   2.41633004e-01   2.26028350e-02
    1.05527180e-01  -3.16032026e-02   3.56926513e-01   1.51980885e-01
    2.76101639e-01   6.73887479e-02   5.03438625e-02   5.87043804e-02
    2.17410195e-01   1.90769246e-01   1.66275222e-01  -2.25387182e-02
   -4.07006441e-02   1.86523497e-02   3.55593965e-02   7.67098774e-02
    5.41168982e-02   1.82997954e-03  -1.06387814e-01   3.21923154e-02
   -5.03033207e-02   2.57983104e-01]
 [  1.05902102e-01   4.73844545e-01   5.37487494e-02  -7.20638781e-03
    4.84014835e-02   6.66524068e-02   4.53663481e-02   3.17864061e-02
   -3.48308747e-01  -1.42316297e-01   2.06736652e-01  -1.59192377e-01
   -1.16708078e-01  -1.30026185e-02  -4.34870459e-02   1.87096064e-01
    8.54676853e-02   1.03314831e-02  -1.51788920e-01  -2.58569260e-01
    7.07110772e-02  -1.35307492e-01   1.10443628e-03  -9.18773088e-02
   -7.20122232e-02   4.33767778e-01   3.75349831e-01  -2.22521778e-02
    1.29157960e-02   1.64834697e-01]
 [ -3.26935606e-01   4.45168360e-01  -2.30866221e-01   5.59195533e-02
    1.24083842e-01   9.84642122e-02   2.19909398e-01   2.88787430e-03
   -5.18013866e-02   3.06423549e-01   2.16240945e-01   2.64952190e-02
    4.29749373e-02  -5.55173866e-02   4.05826204e-02   6.74128873e-02
    1.12391739e-01  -3.37610856e-02  -2.82940804e-02  -6.31182333e-03
    5.82471614e-03  -4.07143250e-02   5.12738420e-02   1.89204055e-01
    2.97420842e-01   2.15003670e-02  -5.10709967e-01   5.56078539e-02
    2.58145605e-02  -4.06763771e-02]]
```

In [37]:

```
transform_data = pca.fit_transform(close_prices)
type(transform_data)
```

Out[37]:

```
numpy.ndarray
```

In [23]:

```
# Вычислите коэффициент корреляции Пирсона между значением по первой компоненте и и
from numpy import corrcoef

first_column = [item[0] for item in transform_data]
corrcoef(first_column, djia_index['^DJI'])
# answer 2: (При необходимости округляйте дробную часть до двух знаков.)  0.91
```

Out[23]:

```
array([[ 1.        ,  0.90965222],
       [ 0.90965222,  1.        ]])
```

In [7]:

```
# Какая компания имеет наибольший вес в первой компоненте?
# Какая компания имеет наибольший вес в первой компоненте? Укажите ее название с бо
first = list(pca.components_[0])
first.index(max(first))
```

Out[7]:

```
26
```

In [54]:

```
close_prices = pandas.read_csv(filepath_or_buffer='/Volumes/fast64/My/MachineLearni
close_prices[[27]] # because first column is "data"
# answer 3:  V - Visa
```

Out[54]:

	V
0	196.240005
1	193.339996
2	191.559998
3	193.559998
4	193.050003
5	191.100006
6	193.220001
7	191.820007
8	188.649994
9	190.479996
10	186.330002
11	182.529999

12	183.850006
13	189.029999
14	192.199997
15	193.440002
16	191.369995
17	195.630005
18	198.279999
19	200.449997
20	200.039993
21	199.979996
22	198.889999
23	202.910004
24	203.059998
25	203.080002
26	204.240005
27	203.820007
28	196.669998
29	199.160004
...	...
344	264.890015
345	271.799988
346	267.420013
347	265.459991
348	264.549988
349	265.989990
350	270.910004
351	269.630005
352	270.869995
353	269.119995
354	269.100006
355	273.000000

356	273.010010
357	272.470001
358	273.010010
359	273.750000
360	271.309998
361	278.290009
362	275.299988
363	273.750000
364	274.130005
365	269.339996
366	271.420013
367	265.200012
368	264.750000
369	269.579987
370	265.029999
371	269.019989
372	264.500000
373	267.670013

374 rows × 1 columns

```
In [ ]:
```