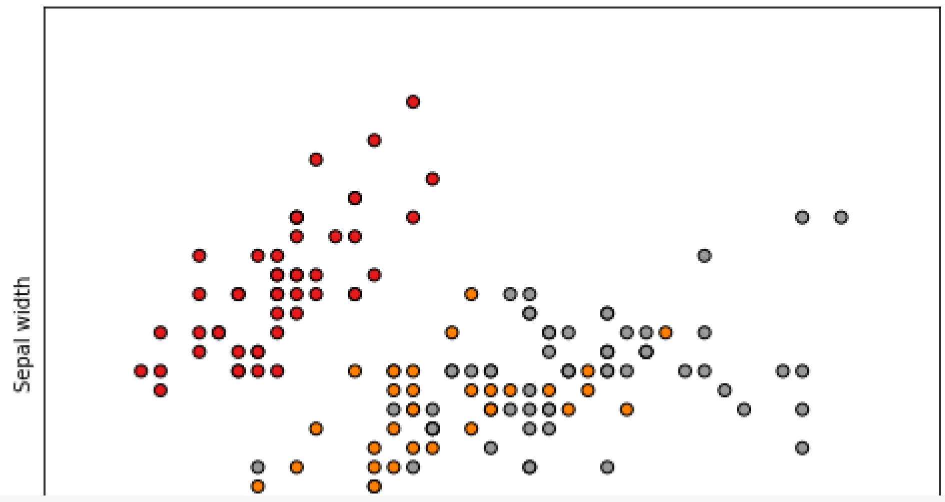
Principal Component Analysis including Cumulative Explained Variance (Iris and MNIST datasets)

```
# Code source: Gaël Varoquaux
# Modified for documentation by Jaques Grobler
# License: BSD 3 clause

import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
from sklearn import datasets
from sklearn.decomposition import PCA

# import some data to play with
iris = datasets.load_iris()
X = iris.data[:, :2]  # we only take the first two features.
y = iris.target

x_min, x_max = X[:, 0].min() - .5, X[:, 0].max() + .5
y_min, y_max = X[:, 1].min() - .5, X[:, 1].max() + .5

plt.figure(2, figsize=(8, 6))
plt.clf()

# Plot the training points
plt.scatter(X[:, 0], X[:, 1], c=y, cmap=plt.cm.Set1,
            edgecolor='k')
plt.xlabel('Sepal length')
plt.ylabel('Sepal width')

plt.xlim(x_min, x_max)
plt.ylim(y_min, y_max)
plt.xticks(())
plt.yticks(())

# To getter a better understanding of interaction of the dimensions
# plot the first three PCA dimensions
fig = plt.figure(1, figsize=(8, 6))
ax = Axes3D(fig, elev=-150, azim=110)
X_reduced = PCA(n_components=3).fit_transform(iris.data)
ax.scatter(X_reduced[:, 0], X_reduced[:, 1], X_reduced[:, 2], c=y,
           cmap=plt.cm.Set1, edgecolor='k', s=40)
ax.set_title("First three PCA directions")
ax.set_xlabel("1st eigenvector")
ax.w_xaxis.set_ticklabels([])
ax.set_ylabel("2nd eigenvector")
ax.w_yaxis.set_ticklabels([])
ax.set_zlabel("3rd eigenvector")
ax.w_zaxis.set_ticklabels([])

plt.show()
```



```
from sklearn.datasets import fetch_openml
mnist = fetch_openml('mnist_784')
```

```
from sklearn.model_selection import train_test_split
# test_size: what proportion of original data is used for test set
train_img, test_img, train_lbl, test_lbl = train_test_split( mnist.data, mnist.target, test_size=1/7.0, rando
```

```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
# Fit on training set only.
scaler.fit(train_img)
# Apply transform to both the training set and the test set.
train_img = scaler.transform(train_img)
test_img = scaler.transform(test_img)
```

```
from sklearn.decomposition import PCA
# Make an instance of the Model
pca = PCA(.95)
```

```
pca.fit(train_img)
```

```
PCA(copy=True, iterated_power='auto', n_components=0.95, random_state=None,
    svd_solver='auto', tol=0.0, whiten=False)
```

```
train_img = pca.transform(train_img)
test_img = pca.transform(test_img)
```

```
from sklearn.linear_model import LogisticRegression
```

```
# all parameters not specified are set to their defaults
# default solver is incredibly slow which is why it was changed to 'lbfgs'
logisticRegr = LogisticRegression(solver = 'lbfgs')
```

```
logisticRegr.fit(train_img, train_lbl)
```

```
/usr/local/lib/python3.6/dist-packages/sklearn/linear_model/_logistic.py:940: ConvergenceWarning: lbfgs faile
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  extra_warning_msg=_LOGISTIC_SOLVER_CONVERGENCE_MSG)
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
                   intercept_scaling=1, l1_ratio=None, max_iter=100,
                   multi_class='auto', n_jobs=None, penalty='l2',
                   random_state=None, solver='lbfgs', tol=0.0001, verbose=0,
                   warm_start=False)
```

```
# Predict for One Observation (image)
logisticRegr.predict(test_img[0].reshape(1,-1))
```

```
array(['0'], dtype=object)
```

```
# Predict for One Observation (image)
logisticRegr.predict(test_img[0:10])
```

```
array(['0', '4', '1', '2', '4', '7', '7', '1', '1', '7'], dtype=object)
```

```
print('Accuracy Score: %0.3f' % logisticRegr.score(test_img, test_lbl))
```

```python
print('Estimated number of components: \n', (pca.n_components_))
```

```
Estimated number of components: 
 327
```

```python
print('Amount of variance explained by each of the selected components: \n', (pca.explained_variance_))
```

```
Amount of variance explained by each of the selected components: 
 [40.59415525 29.01680964 26.87224996 20.86433884 18.05674875 15.74196929
 13.77127266 12.54612494 11.00094033 10.03492632  9.64585917  8.64956409
  7.9968328   7.8424034   7.3814671   7.16859604  6.68823549  6.60819348
  6.40138235  6.2331149   5.91955064  5.73838634  5.4859824   5.32445389
  5.14807896  4.9699168   4.92104344  4.75373005  4.51349763  4.41052839
  4.31141958  4.23396877  4.0886635   4.07135857  4.04759614  3.98510077
  3.83642443  3.80463967  3.70089599  3.65235589  3.47094549  3.41322949
  3.39367785  3.27195748  3.21237257  3.20305992  3.15633079  3.14508318
  3.08792167  3.07664551  2.98364544  2.88352378  2.86964419  2.8042245
  2.78942547  2.766402    2.68205915  2.6503738   2.630463    2.59839835
  2.53415884  2.49134621  2.4813325   2.45818633  2.4338657   2.40472015
  2.37029837  2.30997244  2.28731369  2.25720798  2.2316704   2.20791417
  2.19627765  2.17658648  2.1622007   2.13263816  2.11856585  2.09915978
  2.07887268  2.06769352  2.04099226  2.03129133  2.01085963  2.00592863
  2.00189849  1.99304477  1.97483524  1.9633447   1.93452927  1.91554913
  1.9036724   1.88455667  1.87426626  1.87028656  1.85508397  1.81058262
  1.80477181  1.79400846  1.76726368  1.75065918  1.72635723  1.70760674
  1.69206454  1.65908045  1.64486947  1.61970796  1.60841903  1.59814954
  1.58567607  1.55547553  1.5540458   1.54411919  1.53421886  1.51709899
  1.4838185   1.47340942  1.46966777  1.45220574  1.44236705  1.41238479
  1.40454052  1.39277132  1.3734908   1.37171373  1.36062348  1.35284157
  1.34119435  1.32909986  1.31466917  1.29719283  1.28413076  1.27598694
  1.25762271  1.25412644  1.22615998  1.2037214   1.20262191  1.19642729
  1.18147374  1.17994621  1.16277097  1.15244952  1.13700231  1.12554587
  1.1221124   1.11309523  1.10683259  1.10497361  1.08577893  1.07759413
  1.06398967  1.0607537   1.05401173  1.04887655  1.03275367  1.0286867
  1.02412814  1.02000472  1.00950214  1.00724437  1.00169961  0.99974496
  0.99888706  0.99578642  0.98456272  0.98066593  0.97476626  0.96868066
  0.9647658   0.96287354  0.95242638  0.93805526  0.9347652   0.9291984
  0.92557676  0.91081254  0.90771945  0.89715684  0.89321422  0.87778269
  0.87064796  0.86800266  0.85784502  0.85514628  0.84663193  0.84059328
  0.83311669  0.82824188  0.82113901  0.80726691  0.80012303  0.7987441
  0.79259406  0.7745348   0.76589166  0.757824    0.75413251  0.74949348
  0.74610482  0.74080133  0.73195497  0.7210011   0.71744195  0.70543181
  0.70182957  0.69836981  0.69548244  0.69272035  0.6812968   0.67814608
  0.67127001  0.66761094  0.65178243  0.64876063  0.64320378  0.63619284
  0.63160897  0.62538878  0.62459465  0.61452465  0.61254544  0.60556023
  0.6003223   0.59550632  0.59229828  0.58819919  0.5784161   0.57667491
  0.57094249  0.56848965  0.56353296  0.56040965  0.55743955  0.55105236
  0.5446728   0.5425334   0.5314415   0.5301174   0.52234868  0.52065704
  0.51543481  0.50432032  0.50061778  0.49998156  0.49465156  0.49199992
  0.49122325  0.48934319  0.48471085  0.47606919  0.47046353  0.46928044
  0.46416894  0.45827886  0.45619664  0.4541791   0.44896345  0.44707118
  0.44345598  0.4393925   0.43652367  0.43274673  0.43109523  0.42784114
  0.42133268  0.41800004  0.41717605  0.4120701   0.40959595  0.40696498
  0.40040754  0.39755227  0.39582781  0.39314513  0.38960975  0.38654351
  0.38612122  0.38208062  0.38021094  0.37561398  0.36962957  0.36800565
  0.36585549  0.36334019  0.36149871  0.35950529  0.35771871  0.35363913
  0.35061089  0.34742429  0.34539619  0.34416414  0.34164758  0.33387059
  0.33185349  0.32878938  0.32327587  0.31870139  0.31658394  0.31555253
  0.31172425  0.31129761  0.3101868   0.30903872  0.30359758  0.30122891
  0.29892752  0.29772821  0.29447653  0.29377227  0.29051022  0.28788908
  0.28345726  0.28217358  0.27989973  0.27796788  0.27526011  0.27349215
  0.27298909  0.27008938  0.26892642  0.26689507  0.266189    0.26287416
  0.26188522  0.25859412  0.25807053]
```

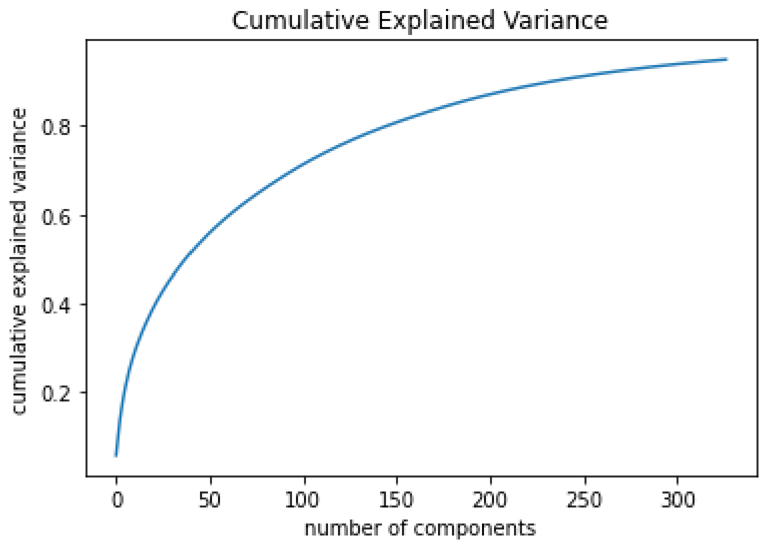
```python
print('Principal axis representing the directions of maximum variance: \n', (pca.components_))
```

```
Principal axis representing the directions of maximum variance: 
 [[-1.49613708e-19 -2.77555756e-17 -0.00000000e+00 ... -0.00000000e+00
  -0.00000000e+00 -0.00000000e+00]
 [ 5.47306713e-19  2.77555756e-16  1.11022302e-16 ... -0.00000000e+00
  -0.00000000e+00 -0.00000000e+00]
 [-1.16585286e-19 -5.55111512e-17 -1.38777878e-17 ... -0.00000000e+00
  -0.00000000e+00 -0.00000000e+00]
 ...
 [ 1.15162705e-18  9.54097912e-17  6.59194921e-17 ...  0.00000000e+00
   0.00000000e+00  0.00000000e+00]
 [-1.78512164e-18 -6.07153217e-17  6.07153217e-17 ...  0.00000000e+00
   0.00000000e+00  0.00000000e+00]
 [-1.17426323e-19 -1.40295761e-16 -9.29161262e-17 ...  0.00000000e+00
   0.00000000e+00  0.00000000e+00]]
```

```python
print('Percentage variance explained by each of the selected components: \n', pca.explained_variance_ratio_)
```

```
Percentage variance explained by each of the selected components:
 [0.05685361 0.04063911 0.03763558 0.02922128 0.02528914 0.02204721
 0.01928718 0.01757131 0.01540722 0.01405428 0.01350938 0.01211403
 0.01119986 0.01098358 0.01033802 0.01003988 0.00936712 0.00925502
 0.00896537 0.00872971 0.00829055 0.00803682 0.00768332 0.00745709
 0.00721007 0.00696055 0.0068921  0.00665777 0.00632132 0.00617711
 0.0060383  0.00592983 0.00572632 0.00570209 0.00566881 0.00558128
 0.00537305 0.00532854 0.00518324 0.00511526 0.00486119 0.00478035
 0.00475297 0.0045825  0.00449905 0.004486   0.00442056 0.0044048
 0.00432475 0.00430896 0.00417871 0.00403848 0.00401904 0.00392742
 0.00390669 0.00387445 0.00375632 0.00371195 0.00368406 0.00363915
 0.00354918 0.00348922 0.0034752  0.00344278 0.00340872 0.0033679
 0.00331969 0.0032352  0.00320347 0.0031613  0.00312554 0.00309227
 0.00307597 0.00304839 0.00302824 0.00298684 0.00296713 0.00293995
 0.00291154 0.00289588 0.00285848 0.0028449  0.00281628 0.00280938
 0.00280373 0.00279133 0.00276583 0.00274974 0.00270938 0.0026828
 0.00266616 0.00263939 0.00262498 0.00261941 0.00259811 0.00253579
 0.00252765 0.00251258 0.00247512 0.00245186 0.00241783 0.00239157
 0.0023698  0.0023236  0.0023037  0.00226846 0.00225265 0.00223827
 0.0022208  0.0021785  0.0021765  0.0021626  0.00214873 0.00212475
 0.00207814 0.00206356 0.00205832 0.00203387 0.00202009 0.0019781
 0.00196711 0.00195063 0.00192362 0.00192114 0.0019056  0.0018947
 0.00187839 0.00186145 0.00184124 0.00181677 0.00179847 0.00178707
 0.00176135 0.00175645 0.00171728 0.00168586 0.00168432 0.00167564
 0.0016547  0.00165256 0.0016285  0.00161405 0.00159241 0.00157637
 0.00157156 0.00155893 0.00155016 0.00154756 0.00152067 0.00150921
 0.00149016 0.00148562 0.00147618 0.00146899 0.00144641 0.00144071
 0.00143433 0.00142855 0.00141384 0.00141068 0.00140292 0.00140018
 0.00139898 0.00139464 0.00137892 0.00137346 0.0013652  0.00135667
 0.00135119 0.00134854 0.00133391 0.00131378 0.00130917 0.00130138
 0.0012963  0.00127563 0.00127129 0.0012565  0.00125098 0.00122937
 0.00121937 0.00121567 0.00120144 0.00119766 0.00118574 0.00117728
 0.00116681 0.00115998 0.00115004 0.00113061 0.0011206  0.00111867
 0.00111006 0.00108476 0.00107266 0.00106136 0.00105619 0.00104969
 0.00104495 0.00103752 0.00102513 0.00100979 0.0010048  0.00098798
 0.00098294 0.00097809 0.00097405 0.00097018 0.00095418 0.00094977
 0.00094014 0.00093501 0.00091285 0.00090861 0.00090083 0.00089101
 0.00088459 0.00087588 0.00087477 0.00086066 0.00085789 0.00084811
 0.00084077 0.00083403 0.00082954 0.00082379 0.00081009 0.00080765
 0.00079963 0.00079619 0.00078925 0.00078487 0.00078071 0.00077177
 0.00076283 0.00075984 0.0007443  0.00074245 0.00073157 0.0007292
 0.00072189 0.00070632 0.00070113 0.00070024 0.00069278 0.00068906
 0.00068798 0.00068534 0.00067886 0.00066675 0.0006589  0.00065724
 0.00065009 0.00064184 0.00063892 0.00063609 0.00062879 0.00062614
 0.00062108 0.00061539 0.00061137 0.00060608 0.00060376 0.00059921
 0.00059009 0.00058542 0.00058427 0.00057712 0.00057365 0.00056997
 0.00056079 0.00055679 0.00055437 0.00055061 0.00054566 0.00054137
 0.00054078 0.00053512 0.0005325  0.00052606 0.00051768 0.00051541
 0.00051239 0.00050887 0.00050629 0.0005035  0.000501   0.00049528
 0.00049104 0.00048658 0.00048374 0.00048201 0.00047849 0.0004676
 0.00046477 0.00046048 0.00045276 0.00044635 0.00044339 0.00044194
 0.00043658 0.00043598 0.00043443 0.00043282 0.0004252  0.00042188
 0.00041866 0.00041698 0.00041243 0.00041144 0.00040687 0.0004032
```

```
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
plt.plot(np.cumsum(pca.explained_variance_ratio_))
plt.title('Cumulative Explained Variance')
plt.xlabel('number of components')
plt.ylabel('cumulative explained variance');
```


Cumulative Explained Variance
cumulative explained variance
number of components