Machine Learning Lab 10

Dimensionality Reduction

Dimensionality reduction is the process of reducing the number of random variables under consideration, by obtaining a set of principal variables. It can be divided into feature selection and feature extraction. There are two components of dimensionality reduction, the first one being feature selection, in which, we try to find a subset of the original set of variables, or features, to get a smaller subset which can be used to model the problem. It is usually done using filtered, wrapped and embedded, the second one is feature extraction. This reduces the data in a high dimensional space to a lower dimension space, i.e. a space with lesser no. of dimensions.

In this experiment we use three methods for dimensionality reduction namely:

* Principal Component Analysis (PCA)

* Linear Discriminant Analysis (LDA)

* Kernel Principal Component Analysis (KPCA)

The dataset

The dataset used to perform this experiment is the wine quality dataset, it is a combination of data on two types of wine variants, namely red wine and white wine, of the portuguese “Vinho Verde” wine. The dataset contains information on the parameters for fixed acidity, volatile acidity, citric acid, residual sugar, chlorides, free sulfur dioxide, total sulfur dioxide, density, pH, sulphates, alcohol.

Experiment

In this experiment I used the sklearn’s PCA(principal component analysis), LDA(linear discriminant analysis) and KPCA algorithms to predict the quality of a wine.

Using the pandas library in I loaded the red wine and white wine datasets into the memory from their respective csv files and then merged the two datasets into one single pandas dataframe. Then I bin the data points into good and bad wine quality data points this is done to reduce the skewness of the dataset, with the original quality classes (i.e. from 3 to 9 the dataset was very skewed). The criterion used for binning is the wine quality. A quality of above and equal to 6 is considered good, where as that below 6 is bad. Thus instead of 6 now I am left with just two categories in my dataset.

Using the standard scaler in sklearn I scaled the all the features of the dataset so as to avoid any feature to dominate while the algorithm is learning on the data.

For performing the experiment I started applying just a logistic regression to my dataset followed by applying a PCA and then applying a logistic regression again. And

after applying the PCA, there was a 0.02 increase in the score of the logistic regression model from 0.96 to 0.98 on the training set.

Next I used a scatter plot to visualize the dataset with reduced dimensions and from the graph it is evident that with reduced dimensions the data points corresponding to the good wine and bad wine samples cluster together.

Next I reduced my data with the LDA algorithm however, with my data it always reduced the dimensions to 1 no matter what, and hence it wasn't useful for any interpretations in my case.

Finally I applied the KPCA with rbf (Radial Basis Function) as the kernel for my PCA and then used a scatter plot to visualize the data and the results were similar, the similar data points clustered together.

This is because once the dimensionality is reduced, the new dimensions contain much more information as compared to the previous dimensions and thus are much better representative of the data than randomly selecting any same number of dimensions from the data. Thus we observe the clustering when we plot the data.

The code and plots can be found in the accompanying jupyter notebook.

PCA + LDA + KPCA

November 1, 2018

1 Lab 10

2 Dimensionality Reduction

2.1 Submitted to: Prof. Sweetlin Hemlatha

2.2 Submitted by: Prateek Singh (15BCE1091)

In [2]:
```
import random
import numpy as np
import pandas as pd
import seaborn as sn
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import precision_score, recall_score
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA, KernelPCA
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis

import matplotlib.pyplot as plt
from matplotlib.colors import ListedColormap
%matplotlib inline
```

In [3]:
```
sn.set(style='ticks', color_codes=True, font_scale=1)

white_wine = pd.read_csv('../Dataset/winequality-white.csv', sep=';')
white_wine['wine_type'] = 0
red_wine = pd.read_csv('../Dataset/winequality-red.csv', sep=';')
red_wine['wine_type'] = 1

wine_data = pd.concat([white_wine, red_wine])
wine_data.tail()
```

Out[3]:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	\
1594	6.2	0.600	0.08	2.0	0.090	
1595	5.9	0.550	0.10	2.2	0.062	
1596	6.3	0.510	0.13	2.3	0.076	
1597	5.9	0.645	0.12	2.0	0.075	

```
1598            6.0             0.310         0.47             3.6      0.067

      free sulfur dioxide  total sulfur dioxide  density    pH  sulphates  \
1594                 32.0                  44.0  0.99490  3.45       0.58
1595                 39.0                  51.0  0.99512  3.52       0.76
1596                 29.0                  40.0  0.99574  3.42       0.75
1597                 32.0                  44.0  0.99547  3.57       0.71
1598                 18.0                  42.0  0.99549  3.39       0.66

      alcohol  quality  wine_type
1594     10.5        5          1
1595     11.2        6          1
1596     11.0        6          1
1597     10.2        5          1
1598     11.0        6          1
```

In [4]:
```
scaler = StandardScaler()

data = wine_data.iloc[:,:11].values
scaled_features = scaler.fit_transform(data)

wine_data_scaled = pd.DataFrame(scaled_features, index=wine_data.index, columns=wine_da
wine_data_scaled.head()
```

Out[4]:
```
   fixed acidity  volatile acidity  citric acid  residual sugar  chlorides  \
0      -0.166089         -0.423183     0.284686        3.206929  -0.314975
1      -0.706073         -0.240949     0.147046       -0.807837  -0.200790
2       0.682458         -0.362438     0.559966        0.306208  -0.172244
3      -0.011808         -0.666161     0.009406        0.642523   0.056126
4      -0.011808         -0.666161     0.009406        0.642523   0.056126

   free sulfur dioxide  total sulfur dioxide   density        pH  sulphates  \
0             0.815565              0.959976  2.102214 -1.359049  -0.546178
1            -0.931107              0.287618 -0.232332  0.506915  -0.277351
2            -0.029599             -0.331660  0.134525  0.258120  -0.613385
3             0.928254              1.243074  0.301278 -0.177272  -0.882212
4             0.928254              1.243074  0.301278 -0.177272  -0.882212

    alcohol
0 -1.418558
1 -0.831615
2 -0.328521
3 -0.496219
4 -0.496219
```

In [5]:
```
X_train, X_test, Y_train, Y_test = train_test_split(wine_data_scaled,
                                                    wine_data.iloc[:, 12],
                                                    test_size=0.2,
                                                    random_state=42)
```

Applying logistic regression without any dimensionality reduction

```
In [6]: model_ll = LogisticRegression(penalty='l2', solver='saga', max_iter=10, multi_class='ov
        model_ll.fit(X_train, Y_train)

        print("Train Score: ", model_ll.score(X_train, Y_train))
        print("Test Score: ", model_ll.score(X_test, Y_test), '\n')
```

```
Train Score:  0.9926880892822783
Test Score:  0.9884615384615385
```

```
/home/prateek/anaconda3/envs/dltf/lib/python3.6/site-packages/sklearn/linear_model/sag.py:326:
  "the coef_ did not converge", ConvergenceWarning)
```

Applying PCA over the dataset

```
In [7]: wine_data_vals = wine_data_scaled.values

        pca = PCA(n_components=2, svd_solver='full', random_state=42)
        reduced_wine_data = pca.fit_transform(X_train)
        reduced_wine_data.shape
```

```
Out[7]: (5197, 2)
```

```
In [8]: model_ll = LogisticRegression(penalty='l2', solver='saga', max_iter=10000, multi_class=
        model_ll.fit(reduced_wine_data, Y_train)
```

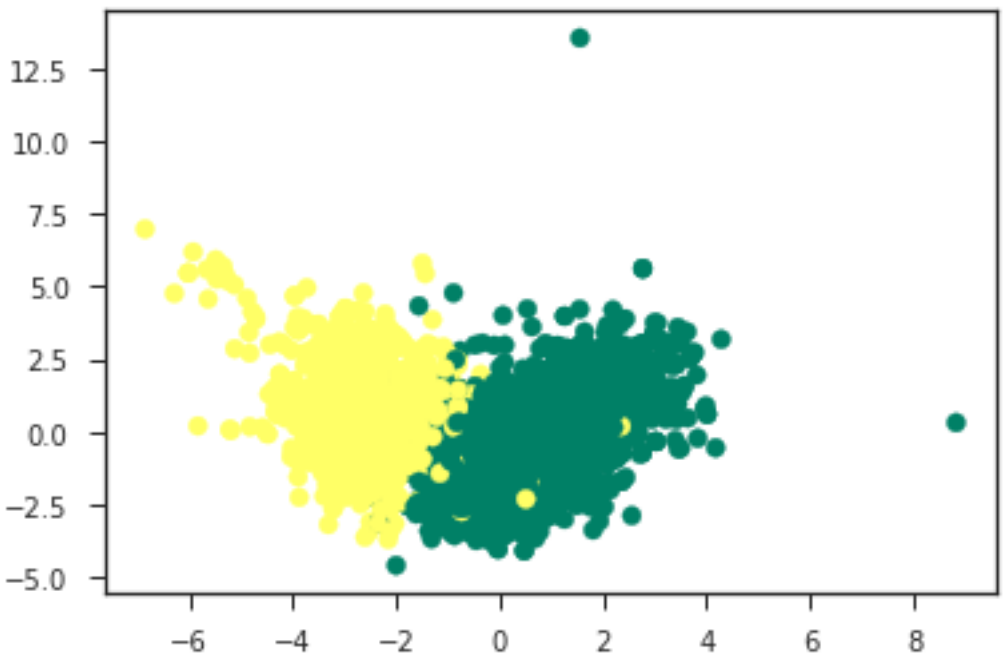
```
Out[8]: LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
                  intercept_scaling=1, max_iter=10000, multi_class='multinomial',
                  n_jobs=1, penalty='l2', random_state=None, solver='saga',
                  tol=0.0001, verbose=0, warm_start=False)
```

```
In [1]: print(model_ll.score(reduced_wine_data, Y_train))
        print(model_ll.score(pca.transform(X_test), Y_test))
```

```
0.9659534346738503
0.9807692307692307
```

```
In [10]: plt.scatter(reduced_wine_data[:, 0], reduced_wine_data[:, 1], c=Y_train, cmap='summer
```

```
Out[10]: <matplotlib.collections.PathCollection at 0x7f442788de48>
```



Applying LDA over the dataset

```
In [11]: lda = LinearDiscriminantAnalysis(solver='eigen', n_components=3)
         wine_data_reduced = lda.fit_transform(X_train, Y_train)

In [12]: wine_data_reduced.shape

Out[12]: (5197, 1)

In [13]: wine_data_reduced

Out[13]: array([[-0.5634886 ],
                [-0.53612547],
                [-0.15322816],
                ...,
                [ 1.34187962],
                [ 1.35577714],
                [-0.22737771]])
```

Applying KPCA on the dataset

```
In [14]: kpca = KernelPCA(n_components=2, kernel='rbf', random_state=42)
         wine_data_reduced = kpca.fit_transform(X_train)

In [15]: print(model_ll.score(reduced_wine_data, Y_train))
         print(model_ll.score(kpca.transform(X_test), Y_test))
```

```
0.9859534346738503
0.7584615384615384
```

```
In [16]: plt.scatter(reduced_wine_data[:, 0], reduced_wine_data[:, 1], c=Y_train, cmap='summer
```

```
Out[16]: <matplotlib.collections.PathCollection at 0x7f4425ac13c8>
```