# CS156 Assignment 3: Dimensionality Reduction

*Yoav Rabinovich, Oct 2018*

---

In [1]:

```
#Collaborated on original preclass work with Michelle and Kalia #Michaliav
from os import listdir, stat
from PIL import Image, ImageFile
import numpy as np
from sklearn.svm import LinearSVC
from sklearn.model_selection import train_test_split
from sklearn.pipeline import Pipeline
from sklearn.decomposition import PCA
from sklearn.metrics import classification_report
from sklearn.model_selection import GridSearchCV
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis as LDA
import matplotlib.pyplot as plt
ImageFile.LOAD_TRUNCATED_IMAGES = True
```

In [2]:

```
def img_to_array(img,size):
    img = img.resize(size)
    img_arr = np.frombuffer(img.tobytes(), dtype=np.uint8)
    return img_arr
```

In [11]:

```
resizing=(60,30)
#extract jerseys with white background to maximize similarity
jerseysdir = listdir("51ShirtsJerseys/Jerseys/")
jerseys = []
#store the exact file sizes for quick duplicate removal without hashing
visited = []
for image in jerseysdir:
    imagepath = "51ShirtsJerseys/Jerseys/" + image
    img = Image.open(imagepath)
    if (img.getpixel((0,0)) == (255,255,255)):
        jerseys.append(img_to_array(img,resizing))
        visited.append(stat(imagepath).st_size)
    img.close()

#extract shirts that aren't also jerseys with white background
shirtsdir = listdir("51ShirtsJerseys/Shirts/")
shirts = []
for image in shirtsdir:
    imagepath = "51ShirtsJerseys/Shirts/" + image
    if stat(imagepath).st_size not in visited:
        img = Image.open(imagepath)
        if (img.getpixel((0,0)) == (255,255,255)):
            shirts.append(img_to_array(img,resizing))
        img.close()
```

In [4]:

```
#OPTIONAL
#take subset to optimize for testing execution speed
#shirts = shirts[:100]
#jerseys = jerseys[:100]
print(len(shirts))
print(len(jerseys))
print(shirts[0].shape)
```

```
309
372
(5400,)
```

In [5]:

```
#extract data
raw_data = [(row,'1') for row in jerseys] + [(row,'0') for row in shirts]
X = np.array([x for (x,y) in raw_data])
y = np.array([y for (x,y) in raw_data])

#train test split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.25)
```

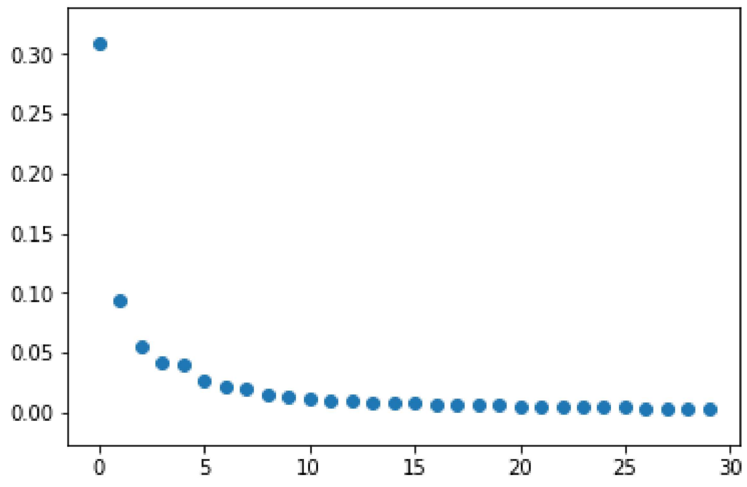
In [6]:

```
#Linear SVM for classification

gs = GridSearchCV(LinearSVC(), {"C": [0.01,0.1,1,10,100]},cv=3, n_jobs=-1, verbose=1)
gs.fit(X_train, y_train)
print(gs.best_params_, gs.best_score_)
print("Train Error:")
print(classification_report(y_train,gs.best_estimator_.predict(X_train)))
print("Test Error:")
print(classification_report(y_test,gs.best_estimator_.predict(X_test)))
```

```
Fitting 3 folds for each of 5 candidates, totalling 15 fits

[Parallel(n_jobs=-1)]: Using backend LokyBackend with 8 concurrent workers.
[Parallel(n_jobs=-1)]: Done  15 out of  15 | elapsed:   25.3s finished
C:\Users\rabin\anaconda3\lib\site-packages\sklearn\svm\base.py:922: Converge
nceWarning: Liblinear failed to converge, increase the number of iterations.
  "the number of iterations.", ConvergenceWarning)

{'C': 100} 0.6647058823529411
Train Error:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00       238
           1       1.00      1.00      1.00       272

   micro avg       1.00      1.00      1.00       510
   macro avg       1.00      1.00      1.00       510
weighted avg       1.00      1.00      1.00       510

Test Error:
              precision    recall  f1-score   support

           0       0.61      0.66      0.64        71
           1       0.74      0.70      0.72       100

   micro avg       0.68      0.68      0.68       171
   macro avg       0.68      0.68      0.68       171
weighted avg       0.69      0.68      0.69       171
```

In [7]:

```
#PCA

#Check for explained variace over n_components
pca = PCA()
pca.fit(X_train)
plt.scatter(list(range(30)), pca.explained_variance_ratio_[:30])
plt.show()
```



In [16]:

```
#Choosing 5 by elbow method
pca=PCA(n_components=5)
pca.fit(X_train)
pca_train = pca.transform(X_train)
SVMclf = LinearSVC()
SVMclf.fit(pca_train, y_train)

print("Train Error:")
print(classification_report(y_train,SVMclf.predict(pca_train)))
print("Test Error:")
print(classification_report(y_test,SVMclf.predict(pca.transform(X_test))))
```

```
Train Error:
              precision    recall  f1-score   support

           0       0.59      0.67      0.63       238
           1       0.67      0.58      0.62       272

   micro avg       0.63      0.63      0.63       510
   macro avg       0.63      0.63      0.63       510
weighted avg       0.63      0.63      0.63       510

Test Error:
              precision    recall  f1-score   support

           0       0.57      0.75      0.65        71
           1       0.77      0.60      0.67       100

   micro avg       0.66      0.66      0.66       171
   macro avg       0.67      0.67      0.66       171
weighted avg       0.69      0.66      0.66       171


C:\Users\rabin\anaconda3\lib\site-packages\sklearn\svm\base.py:922: Converge
nceWarning: Liblinear failed to converge, increase the number of iterations.
  "the number of iterations.", ConvergenceWarning)
```

In [9]:

```
#LDA

pipe = Pipeline([
    ('lda', LDA(n_components=1)),
    ('clf', LinearSVC())
])

param_grid = [
    {
        'clf__C': [0.01,0.1,1,10,100],
    },
]

gs = GridSearchCV(pipe, cv=3, n_jobs=-1, param_grid=param_grid, verbose=1)
gs.fit(X_train, y_train)
print(gs.best_params_, gs.best_score_)
print("Train Error:")
print(classification_report(y_train,gs.best_estimator_.predict(X_train)))
print("Test Error:")
print(classification_report(y_test,gs.best_estimator_.predict(X_test)))
```

```
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 8 concurrent workers.


Fitting 3 folds for each of 5 candidates, totalling 15 fits


[Parallel(n_jobs=-1)]: Done  15 out of  15 | elapsed:    4.2s finished


{'clf__C': 0.01} 0.6490196078431373
Train Error:
              precision    recall  f1-score   support

           0       0.95      0.95      0.95       238
           1       0.95      0.96      0.96       272

   micro avg       0.95      0.95      0.95       510
   macro avg       0.95      0.95      0.95       510
weighted avg       0.95      0.95      0.95       510

Test Error:
              precision    recall  f1-score   support

           0       0.63      0.62      0.62        71
           1       0.73      0.74      0.74       100

   micro avg       0.69      0.69      0.69       171
   macro avg       0.68      0.68      0.68       171
weighted avg       0.69      0.69      0.69       171



C:\Users\rabin\anaconda3\lib\site-packages\sklearn\discriminant_analysis.py:
388: UserWarning: Variables are collinear.
  warnings.warn("Variables are collinear.")
```

In [17]:

```
#PCA and LDA
pipe = Pipeline([
    ('pca', PCA(n_components=5)),
    ('lda', LDA(n_components=1)),
    ('clf', LinearSVC())
])

param_grid = [
    {
        'clf__C': [0.01,0.1,1,10,100],
    },
]

gs = GridSearchCV(pipe, cv=3, n_jobs=-1, param_grid=param_grid, verbose=1)
gs.fit(X_train, y_train)
print(gs.best_params_, gs.best_score_)
print("Train Error:")
print(classification_report(y_train,gs.best_estimator_.predict(X_train)))
print("Test Error:")
print(classification_report(y_test,gs.best_estimator_.predict(X_test)))
```

```
Fitting 3 folds for each of 5 candidates, totalling 15 fits

[Parallel(n_jobs=-1)]: Using backend LokyBackend with 8 concurrent workers.

{'clf__C': 0.01} 0.6843137254901961
Train Error:
              precision    recall  f1-score   support

           0       0.64      0.71      0.67       238
           1       0.72      0.66      0.69       272

   micro avg       0.68      0.68      0.68       510
   macro avg       0.68      0.68      0.68       510
weighted avg       0.68      0.68      0.68       510

Test Error:
              precision    recall  f1-score   support

           0       0.60      0.73      0.66        71
           1       0.78      0.66      0.71       100

   micro avg       0.69      0.69      0.69       171
   macro avg       0.69      0.70      0.69       171
weighted avg       0.71      0.69      0.69       171


[Parallel(n_jobs=-1)]: Done  15 out of  15 | elapsed:    9.0s finished
```

## Analysis

1. Linear Support Vector Machine is a supervised classifier, attempting to linearly separate the data while

minimizing perpendicular distance between points and separator line. Using cross-validation to optimize a penalty parameter proved unfruitful, and the classifier kept on overfitting the data to an extreme amount, classifying correctly 100% of the training data. This is due to the dimensionality of our training data, having many more features than datapoints to train with. Reshaping the pictures to barely recognizable size was not helpful. Considering only pictures with white backgrounds, while scaling back our training set size, proved helpful because such photos are likely to be standardized and less features carry weight in the classification. This is where dimensionality reduction should then have a great effect.

2. Principle Component Analysis is an unsupervised dimensionality reduction method, where data is projected down to lower dimensionality based on the directions of highest variance. Comparing the explained variance over many values for the number of final components, I've used the "elbow method" to pick the point where added dimensionality lead to significantly marginal returns. However, transforming the data and using SVM showed no improvement. However, training accuracy reduced without lowering test accuracy, showing signs of combating overfitting. I was hopeful that considering separability with LDA might improve on the classification accuracy.
3. Linear Discriminant Analysis is a supervised dimensionality reduction method, where data is prohected down to lower dimensionality to maximize difference in means (separability). Sadly, no improvement was gained from LDA either. I took effort to eliminate duplicate data in the shirts and jerseys dataset, but that wasn't the factor limiting the efficacy of separability based classification.

Combining approaches proved promising, and is my official recommendation even though significant improvement wasn't made. Also wear sweaters.

```
In [ ]:
```