PCA Code

```
# Compute a PCA

n_components = 100    ##number of components to retain

pca = PCA(n_components=n_components, whiten=True).fit(X_train)    ##learning PCA decision boundary


# apply PCA transformation

X_train_pca = pca.transform(X_train) ## project the training data into lower dimensional space

X_test_pca = pca.transform(X_test) ## project the test data into lower dimensional space


# train a neural network (using a neural network on the training data)

print("Fitting the classifier to the training set")

clf = MLPClassifier(hidden_layer_sizes=(1024,), batch_size=256, verbose=True, early_stopping=True).fit(X_train_pca, y_train)
```

Example in the link: https://pythonmachinelearning.pro/face-recognition-with-eigenfaces/

LDA Code

```
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis as LDA

lda = LDA(n_components=1) ## calling LDA function and stating the number of components to retain

X_train = lda.fit_transform(X_train, y_train) ## learning LDA decision boundaries and

X_test = lda.transform(X_test) ## projecting the original test data into the transformed space (also called as fisher space)
```

Training a random forest classifier on the data reduced by LDA

```
from sklearn.ensemble import RandomForestClassifier

classifier = RandomForestClassifier(max_depth=2, random_state=0)

classifier.fit(X_train, y_train)

y_pred = classifier.predict(X_test)
```

Example in this link: https://stackabuse.com/implementing-lda-in-python-with-scikit-learn/