Университет ИТМО
Факультет программной инженерии и компьютерной техники

Курс «Искусственный интеллект»

Лабораторная работа № 5

Вариант: 273720

Работу выполнил

Хуан Сыюань
P33101

Преподаватель
Игорь Александрович
Бессмертный

Санкт-Петербург
2021

Задание

Цель: решить задачу многоклассовой классификации, используя в качестве тр енировочного набора данных - набор данных MNIST, содержащий образы рук описных цифр.

1. Используйте метод главных компонент для набора данных MNIST (train dataset объема 60000). Определите, какое минимальное количество главных к омпонент необходимо использовать, чтобы доля объясненной дисперсии пре вышала 0.80 + номер_в_списке % 10. Построить график зависимости доли объ ясненной дисперсии от количества используемых ГК

2. Выведите количество верно классифицированных объектов класса номер_в _списке % 9 для тестовых данных

3. Введите вероятность отнесения 5 любых изображений из тестового набора к назначенному классу

4. Определите Accuracy, Precision, Recall и F1 для обученной модели

5. Сделайте вывод про обученную модель

Выполнение лабораторной работы

Question 1

Codes:

```
[45] dim = 784 # 28*28
     exp_disp = 0.8 + 273720 % 10 / 100
     classa = 273720 % 9
     X_train_ = X_train.reshape(len(X_train), dim)
```

```
[46] from sklearn.decomposition import PCA

     pca = PCA(svd_solver='full')
     pca = pca.fit(X_train_)
```

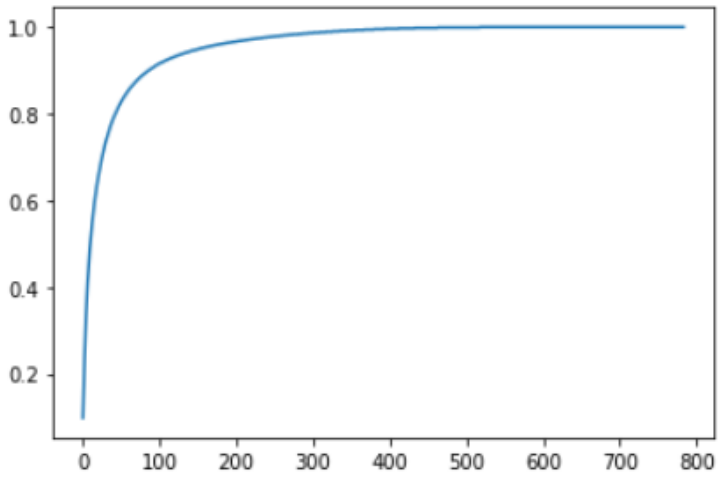
```
[48] explained_variance = np.round(np.cumsum(pca.explained_variance_ratio_),3)
```

```
plt.plot(np.arange(dim), explained_variance, ls = '-')
M = 0
for arg, val in enumerate(np.cumsum(pca.explained_variance_ratio_)):
    if val > exp_disp:
        M = arg + 1
        break
print("The number of Principal Components so that the proportion of the explained variance exceeds " +
          str(exp_disp) + ': ' + str(M))
```

```
X_train = X_train.reshape(len(X_train), dim)
pca = PCA(n_components=M, svd_solver='full')
pca = pca.fit(X_train)
explained_variance = np.round(np.cumsum(pca.explained_variance_ratio_),3)
plt.plot(np.arange(M), explained_variance, ls = '-')
```

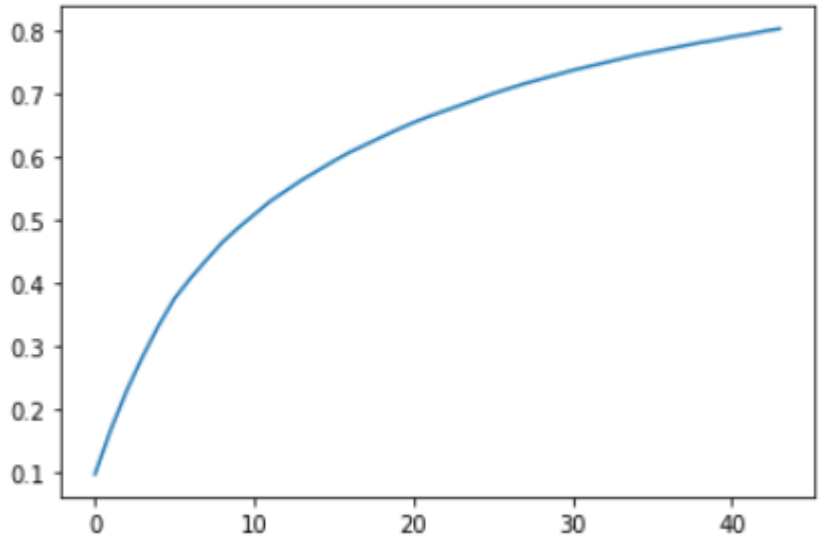
Answers:

```
The number of Principal Components so that the proportion of the explained variance exceeds 0.8: 44
```



Dependence of the proportion of the explained variance on all:

```
[<matplotlib.lines.Line2D at 0x7f4de9c83090>]
```



Question 2

Codes and answers:

```
[58] from sklearn.multiclass import OneVsRestClassifier
     from sklearn.tree import DecisionTreeClassifier
     from sklearn.ensemble import RandomForestClassifier
     from sklearn.model_selection import train_test_split
     from sklearn.metrics import confusion_matrix
```

```
X_train, X_test, y_train, y_test = train_test_split(X_train, y_train, test_size=0.3, random_state=2020)
X_train = pca.transform(X_train)
X_test = pca.transform(X_test)

model1PCA = pca.fit(X_test)
X_test = model1PCA.transform(X_test)

tree = RandomForestClassifier(criterion='gini', min_samples_leaf=10, max_depth=20, n_estimators=10, random_state=2020)
clf = OneVsRestClassifier(tree).fit(X_train, y_train)

y_pred = clf.predict(X_test)

CM = confusion_matrix(y_test, y_pred)

print("The number of correctly classified objects of the class " + str(classa) + ":"
      + str(CM[classa][classa]))
```

```
The number of correctly classified objects of the class 3:233
```

Question 3

Codes and answers:

```
pics = [110, 1014, 996, 520, 250]
for pic in pics:
    print(f"Probability of attribution of picture №{pic} to the assigned class {y_pred[pic]} = {clf.predict_proba(X_test)[pic][y_pred[pic]]}")
```

```
Probability of attribution of picture №110 to the assigned class 0 = 0.5835864097273735
Probability of attribution of picture №1014 to the assigned class 7 = 0.5060023600489468
Probability of attribution of picture №996 to the assigned class 4 = 0.5978781978474554
Probability of attribution of picture №520 to the assigned class 8 = 0.3447542075896968
Probability of attribution of picture №250 to the assigned class 0 = 0.5439224893612968
```

Question 4

Codes and answers:

```
from sklearn.metrics import classification_report, accuracy_score
target_names = ['class 0', 'class 1', 'class 2','class 3','class 4','class 5','class 6','class 7','class 8','class 9']
print("Accuracy: ", accuracy_score(y_test,y_pred))

print(classification_report(y_test, y_pred, target_names = target_names))
```

```
Accuracy:  0.6836111111111111
              precision    recall  f1-score   support

     class 0       0.89      0.86      0.88      1693
     class 1       0.94      0.81      0.87      2075
     class 2       0.46      0.58      0.51      1763
     class 3       0.70      0.79      0.74      1873
     class 4       0.78      0.78      0.78      1756
     class 5       0.50      0.45      0.48      1591
     class 6       0.44      0.37      0.40      1766
     class 7       0.75      0.78      0.77      1886
     class 8       0.64      0.64      0.64      1773
     class 9       0.72      0.71      0.71      1824

    accuracy                           0.68     18000
   macro avg       0.68      0.68      0.68     18000
weighted avg       0.69      0.68      0.68     18000
```

Вывод

As a result of the laboratory work, a model was trained to predict the drawn numbers on the MNIST set using 44 principal components out of 784 available, resulting in a share of explained variance of 0.8 on the test sample. The resulting model has a precision of 0.684 and does a good job of determining the digits 0, 1, 3, 4, 7, and 9 in comparison to the rest of the digits, for which the more informative measures Precision, Recall, and F1 are much lower and less than 0.5.