Министерство науки и высшего образования РФ

Национальный исследовательский университет ИТМО

Факультет Программной инженерии и компьютерных технологий

По дисциплине:
Системы искусственного интеллекта

Лабораторная работа № 5.

«Решение задачи многоклассовой классификации - набор данных MNIST»

Вариант 3

Выполнил: By Минь Хиеу

Группа: P33201

Санкт–Петербург

2022 год.

I. Описание задания

Цель: решить задачу многоклассовой классификации, используя в качестве тренировочного набора данных - набор данных MNIST, содержащий образы рукописных цифр.

1. Используйте метод главных компонент для набора данных MNIST (train dataset объемом 60000). Определите, какое минимальное количество главных компонент необходимо использовать, чтобы доля объясненной дисперсии превышала 0.80 + номер_в_списке % 10. Построить график зависимости доли объясненной дисперсии от количества используемых ГК.
2. Введите 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 или F1 для обученной модели.
5. Сделайте вывод про обученную модель.

Вариант: Номер в списке 3.

II. Выполнение

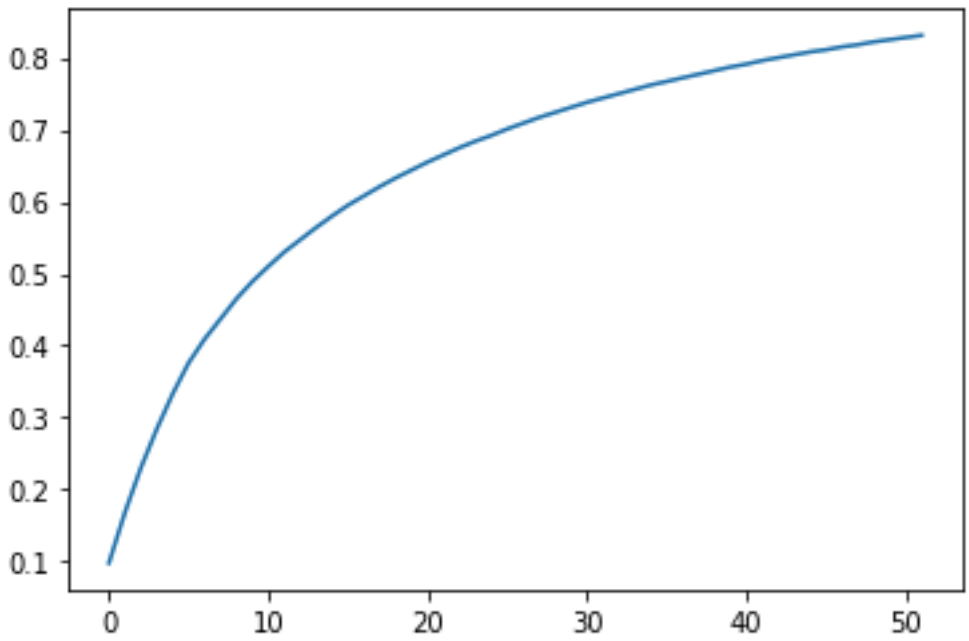
1. Используйте метод главных компонент для набора данных MNIST

Минимальное количество главных компонент необходимо использовать, чтобы оля объясненной дисперсии превышала 0.83.

```
variant_exp = 0.8 + (list_number % 10) / 100
pca = PCA(svd_solver='full')
pca_model = pca.fit(X_train)
X_train_redim = pca_model.transform(X_train)
X_test_redim = pca_model.transform(X_test)
explained_variance = np.round(np.cumsum(pca.explained_variance_ratio_), 3)

components = 0
cum_sum = 0
for i in range(dim):
  components += 1
  cum_sum += pca.explained_variance_ratio_[i]
  if cum_sum >= variant_exp:
    break

print(explained_variance[:components])

print("Explained Variance: " + str(round(cum_sum, 2)))
print("Number of Components: " + str(components))
```

```
[0.098 0.168 0.23  0.284 0.333 0.376 0.409 0.437 0.465 0.489 0.51  0.53
 0.547 0.564 0.58  0.595 0.608 0.621 0.633 0.644 0.655 0.665 0.675 0.684
 0.692 0.701 0.709 0.717 0.724 0.731 0.738 0.744 0.75  0.756 0.762 0.767
 0.772 0.777 0.782 0.787 0.791 0.796 0.8   0.804 0.808 0.811 0.815 0.818
 0.822 0.825 0.828 0.831]
Explained Variance: 0.83
Number of Components: 52
```

График дисперсии:



2. **Введите количество верно классифицированных объектов**

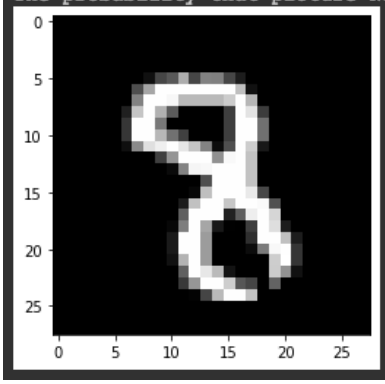
```
y_pred = clf.predict(X_test_redim)

class_variant = list_number % 9
cm = confusion_matrix(y_test, y_pred)

print(f"The number of class {class_variant} images is: ", cm.sum(axis=1)[class_variant])

print("The number of correctly classified images contained in Class " + str(class_variant) + " is: "
      + str(cm[class_variant][class_variant]))
```

```
The number of class 3 images is:  1554
The number of correctly classified images contained in Class 3 is: 1125
```

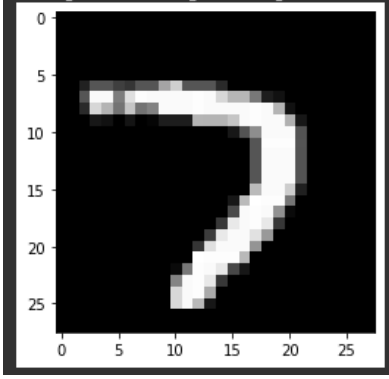
3. **Введите вероятность отнесения 5 любых изображений**

```
num = random.randint(0, 10000)
result = (clf.predict_proba(X_test_redim)[num])[y_pred[num]]
plt.imshow(X_test_show[num], cmap='gray')
print("The probability that picture No." + str(num) + " belongs to Class " +
      str(y_pred[num]) + " is: " + str(round(result, 3)))
```

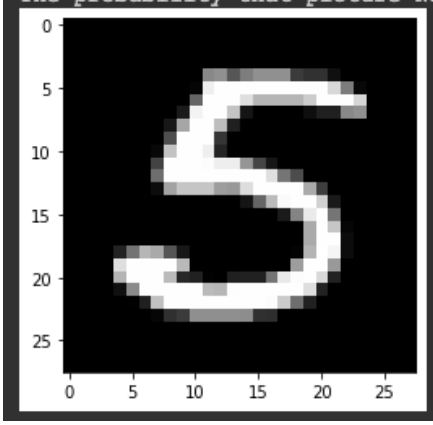
```
The probability that picture No.9222 belongs to Class 3 is: 0.847
```



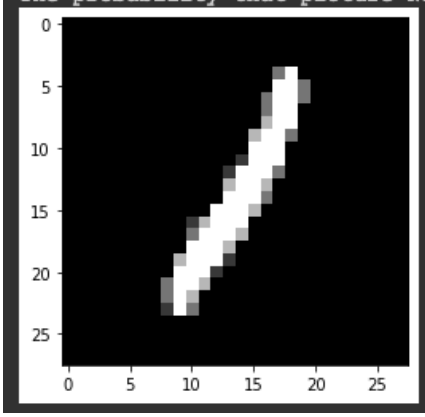

```
The probability that picture No.4803 belongs to Class 7 is: 0.973
```



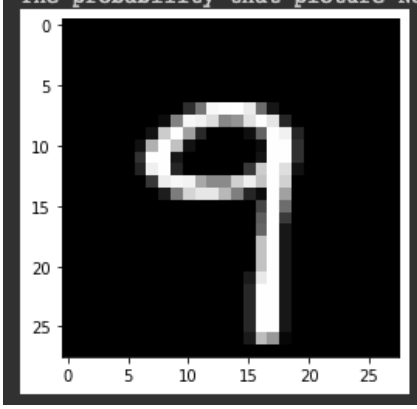
```
The probability that picture No.4665 belongs to Class 5 is: 0.935
```



```
The probability that picture No.9683 belongs to Class 1 is: 0.985
```



```
The probability that picture No.7760 belongs to Class 9 is: 0.892
```



4. **Определите Accuracy, Precision, Recall или F1**

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{P + N} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}$$

Тем не менее, у этой метрики есть одна особенность которую необходимо учитывать. Она присваивает всем документам одинаковый вес, что может быть не корректно в случае если распределение документов в обучающей выборке сильно смещено встрону какого-то одного или нескольких классов.

```
[14] print("Accuracy: " + str(accuracy_score(y_test, y_pred)))

     Accuracy: 0.7876
```

a. **Precision** – точность

$$Precision = \frac{TP}{TP + FP}$$

Точность показывает, какая доля объектов, выделенных классификатором как положительные, действительно является положительными.

b. **Recall** – полнота

$$Recall = \frac{TP}{TP + FN}$$

Полнота показывает, какая часть положительных объектов была выделена классификатором.

c. F1 Score

$$F_1 = \frac{2}{\text{recall}^{-1} + \text{precision}^{-1}} = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} = \frac{\text{tp}}{\text{tp} + \frac{1}{2}(\text{fp} + \text{fn})}$$

Существует несколько способов получить один критерий качества на основе точности и полноты. Один из них — F-мера, гармоническое среднее точности и полноты

```
[15] print(classification_report(y_test, y_pred))
```

	precision	recall	f1-score	support
0	0.93	0.81	0.86	1471
1	0.89	0.94	0.92	1702
2	0.77	0.77	0.77	1537
3	0.79	0.72	0.76	1554
4	0.81	0.75	0.78	1457
5	0.67	0.70	0.68	1356
6	0.83	0.82	0.83	1505
7	0.86	0.85	0.86	1538
8	0.62	0.72	0.67	1442
9	0.72	0.75	0.73	1438
accuracy			0.79	15000
macro avg	0.79	0.78	0.79	15000
weighted avg	0.79	0.79	0.79	15000

III. Вывод

В лабе я выполнил мультиклассовую классификацию с помощью дерева решений на наборе данных рукописных цифр, научился видеть параметры обученной модели — это основные навыки, необходимые для развития в области ML. Классификационная модель дает относительно хорошие результаты. Особенно хорошо распознает такие цифры, как 0, 1.