Практическое задание №1.6.4

Для выполнения практического задания мы будем использовать данные о диабете у женщин. Данные хранятся в файле «diabetes.csv».

Источник данных: https://www.kaggle.com/uciml/pima-indians-diabetes-database

Данные содержат следующие характеристики:

1. Pregnancies – число случаев беременности
2. Glucose – концентрация глюкозы в крови
3. BloodPressure – артериальное диастолическое давление (мм рт. ст.)
4. SkinThickness – толщина кожной складки трехглавой мышцы (мм)
5. Insulin – 2-х часовой сывороточный инсулин
6. BMI – индекс массы тела
7. DiabetesPedigreeFunction – числовой параметр наследственности диабета
8. Age – возраст

Outcome – **целевая переменная**: 1 – наличие заболевания, 0 – отсутствие

Задача: построить модель, определяющую есть или нет заболевание по набору характеристик 1-8.

1. Загрузите данные в DataFrame с помощью функции `read_csv` библиотеки pandas.
2. Как наблюдения (объекты) распределились по классам? Сколько наблюдений в каждом классе? Для ответа на вопрос используйте метод `value_counts()`.
3. Разделите данные на признаки и ответы, а затем на обучающую и тестовую выборки.
4. Обучите **линейную** SVM-модель с помощью класса SVC из sklearn.svm:

```
from sklearn.svm import SVC
svm = SVC(kernel ='linear')
```

5. Оцените качество модели на тестовой выборке. Используйте для этого функцию classification_report. *Что можно сказать о модели?*
6. Стандартизируйте данные и постройте модель на стандартизированных данных. Используйте для стандартизации класс StandardScaler. Оцените качество модели с помощью classification_report. *Улучшилась ли модель?*
7. *Помните ли вы такой способ оценки качества модели как перекрестная проверка?* Воспользуйтесь перекрестной проверкой, чтобы оценить качество моделей. Используйте функцию cross_val_score.

По умолчанию `cross_val_score` использует ту метрику, которую использует сама модель в методе `score`. С помощью параметра `scoring` можно установить другую метрику (список метрик: https://scikit-learn.org/stable/modules/model_evaluation.html#scoring-parameter, столбец Scoring). *Какой вывод можно сделать?*

На следующих страницах представлен программный код к каждому заданию.

1. Загрузите данные в DataFrame с помощью функции `read_csv` библиотеки pandas.

```
import pandas as pd
df = pd.read_csv('diabetes.csv')
```

2. Как наблюдения (объекты) распределились по классам? Сколько наблюдений в каждом классе?

```
df['Outcome'].value_counts()
```

3. Разделите данные на признаки и ответы, а затем на обучающую и тестовую выборки.

```
X = df.drop('Outcome', axis=1)
y = df['Outcome']

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.3, random_state=42)
```

4. Обучите **линейную** SVM-модель с помощью класса `SVC` из sklearn.svm.

```
from sklearn.svm import SVC # Импортируем SVC
svm = SVC(kernel = 'linear') # Линейная модель
svm.fit(X_train, y_train) # Обучаем модель
```

5. Оцените качество модели на тестовой выборке. Используйте для этого функцию `classification_report`.

```
from sklearn.metrics import classification_report

y_pred = svm.predict(X_test) # Делаем предсказание на тестовой выборке
print(classification_report(y_test, y_pred)) # Оцениваем качество
```

Что можно сказать о модели?

6. Стандартизируйте данные и постройте модель на стандартизированных данных. *Улучшилась ли модель?*

```
from sklearn.preprocessing import StandardScaler

scale = StandardScaler() # Создаем экземпляр класса
X_scale = scale.fit_transform(X) # Преобразуем данные
```

```
X_train, X_test, y_train, y_test = train_test_split(X_scale, y,
test_size=0.3, random_state=42)

svm_1 = SVC(kernel = 'linear')
svm_1.fit(X_train, y_train) # Обучаем модель

y_pred = svm_1.predict(X_test) # Делаем предсказание на тестовой выборке
print(classification_report(y_test, y_pred)) # Оцениваем
```

7. *Помните ли вы такой способ оценки качества модели как перекрестная проверка?* Воспользуемся перекрестной проверкой, чтобы оценить качество моделей. Будем использовать уже знакомую функцию `cross_val_score`. По умолчанию `cross_val_score` использует ту метрику, которую использует сама модель в методе `score`. С помощью параметра `scoring` можно установить другую метрику (список метрик: https://scikit-learn.org/stable/modules/model_evaluation.html#scoring-parameter, столбец Scoring).
Для примера возьмем метрику `f1_weighted`:

```
svm_2 = SVC(kernel = 'linear')

from sklearn.model_selection import cross_val_score
cross = cross_val_score(svm_2, X, y, cv=6, scoring =
'f1_weighted') # Для нестандартизированных данных
cross_std = cross_val_score(svm_2, X_scale, y, cv=6, scoring =
'f1_weighted') # Для стандартизированных данных

import numpy as np
# Усредним значения
print(np.mean(cross)) # Для модели, обученной на нестандартизированных
данных
print(np.mean(cross_std)) # Для модели, обученной на стандартизированных
данных
```

Какой вывод можно сделать?