Машинное обучение, DS-поток

Домашнее задание 7

Правила:

- Дедлайн **10 апреля 02:00** (по состоянию на момент выдачи). После дедлайна работы не принимаются кроме случаев наличия уважительной причины.
- Выполненную работу нужно отправить на почту `mipt.stats@yandex.ru` , указав тему письма `"[ml] Фамилия Имя - задание 7"` . Квадратные скобки обязательны. Если письмо дошло, придет ответ от автоответчика.
- Прислать нужно ноутбук и его pdf-версию (без архивов). Названия файлов должны быть такими: `7.N.ipynb` и `7.N.pdf` , где `N` - ваш номер из таблицы с оценками.
- Теоретические задачи необходимо оформить в tex/markdown или же прислать фотку в правильной ориентации рукописного решения, **где все четко видно**.
- Решения, размещенные на каких-либо интернет-ресурсах не принимаются. Кроме того, публикация решения в открытом доступе может быть приравнена к предоставлении возможности списать.
- Для выполнения задания используйте этот ноутбук в качестве основы, ничего не удаляя из него.
- Никакой код из данного задания при проверке запускаться не будет.

Баллы за задание:

- Задача 1 - 3 балла
- Задача 2 - 4 балла
- Задача 3 - 3 балла
- Задача 4 - 3 балла
- Задача 5 - 5 баллов
- Задача 5 - 6 баллов

In [ ]:

```
import os

import numpy as np
import scipy.stats as sps
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

from sklearn.pipeline import Pipeline
from sklearn.svm import SVC, SVR
from sklearn.metrics import accuracy_score, mean_absolute_error
from sklearn.model_selection import train_test_split, RandomizedSearchCV
from sklearn.preprocessing import StandardScaler
from sklearn.datasets import make_moons, make_blobs, make_circles

sns.set(font_scale=1.2, palette='Set2')
```

Задача 1.

Рассмотрим следующие функции

- $K(x, z) = \langle x, z \rangle^k$, где $x, z \in \mathcal{X} = \mathbb{R}^2, k \in \mathbb{N}$
- $K(x, z) = \exp\left(-\gamma \|x - z\|^2\right)$, где $x, z \in \mathcal{X} = \mathbb{R}^2, \gamma > 0$

Для этих ядер найдите спрямляющее пространство $\mathcal{H}$ и отображение $\psi : \mathcal{X} \to \mathcal{H}$, для которого верно $K(x, z) = \langle \psi(x), \psi(z) \rangle$. Какой вид имеет классификатор SVM, основанный на этих ядрах?

Задача 2.

Какие из следующих функций являются ядрами?

- $K(x, z) = \cos(x - z)$, где $x, z \in \mathbb{R}$
- $K(x, z) = \cos^2(x - z)$, где $x, z \in \mathbb{R}$
- $K(x, z) = \sin^2(x - z)$ где $x, z \in \mathbb{R}$
- $K(x, z) = \min\{x, z\}$, где $x, z \in \mathbb{R}_+$

Задача 3.

Рассмотрите следующую двумерную выборку

In [ ]:

```
X, Y = make_circles(n_samples=100)
```

Изобразите эти выборки на плоскости, используя `Y` в качестве цвета точек.

In [ ]:

```

```

Рассмотрим ядро $K(x, z) = \langle x, z \rangle^2$. Переведите точки выборки в соответствующее данному ядру спрямляющее пространство и визуализируйте все проекции на пары координат. Рисуйте эти графики в строчку.

In [ ]:

```

```

Добавьте к признакам некоторый сдвиг и повторите те же действия. Что можно наблюдать?

In [ ]:

```

```

Повторите те же действия с RBF-ядром $K(x, z) = \exp\left(-\gamma \|x - z\|^2\right)$, рассмотрев проекции на пары координат спрямляющего пространства, которые соответствуют многочленам степени не более 3 исходного пространства. Для простоты рисуйте графики в виде сетки $k \times k$.

In [ ]:

```

```

Для RBF-ядра повторите те же действия для следующей выборки.

In [ ]:

```
X, Y = make_moons(n_samples=100)
```

In [ ]:

```

```

Сделайте выводы.

Задача 4.

Даны три случайные двумерные выборки для бинарной классификации:

- С линейно разделимыми классами
- С изогнутыми классами в виде лун
- С вложенными друг в друга классами

Код для генерации каждой выборки приводится далее.

In [ ]:

```
random_state = 20200329
n_samples = 500

X, Y = np.zeros((3, n_samples, 2)), np.zeros((3, n_samples))
X[0], Y[0] = make_blobs(n_samples=n_samples, n_features=2,
                        centers=2, random_state=random_state)
X[1], Y[1] = make_moons(n_samples=n_samples,
                        noise=0.2, random_state=random_state)
X[2], Y[2] = make_circles(n_samples=n_samples,
                          noise=0.1, random_state=random_state)
```

Визуализация этих выборок

In [ ]:

```
plt.figure(figsize=(16, 5))

for i, name in enumerate(['Линейно разделимые', 'Изогнутые', 'Вложенные']):
    plt.subplot(1, 3, i+1)
    plt.scatter(X[i, :, 0], X[i, :, 1], c=Y[i], cmap='spring')
    plt.title(name)
plt.show()
```

Для решения задачи вам выдается функция, аналогичная функции с семинара, но с некоторыми изменениями

In [ ]:

```python
def draw_graphics(models, X, Y, X_test, Y_test, point_size=35,
                  ncol=3, margin=False, params=['C']):
    '''
    Визуализирует решающие правила для каждой модели.
    models --- все обученные SVM-модели, которые нужно визуализировать.
    X --- объекты для визуализации (предполагается обучающая выборка)
    Y --- ответы для визуализации (предполагается обучающая выборка)
    X_test --- объекты, на которых необходимо посчитать качество
    Y_test --- соответствующие им ответы.
    point_size --- размер точки на графике
    ncol --- количество колонок у таблицы графиков
    margin --- если True, то визуализируется решающая функция,
                иначе решающее правило
    params --- список параметров SVM, которые нужно напечатать на графике
    '''

    # определение количества строк таблицы графиков в зависимости
    # от количества колонок и количества моделей
    n_rows = (len(models) + ncol-1) // ncol

    plt.figure(figsize=(16, 5*n_rows))
    for i_model, model in enumerate(models):
        plt.subplot(n_rows, 3, i_model+1)

        # Визуализация опорных векторов model.support_vectors_
        plt.scatter(
            model.support_vectors_[:, 0], model.support_vectors_[:, 1],
            edgecolor='black', s=1.5*point_size,
            alpha=0.5, zorder=10, linewidths=7
        )

        # Визуализация остальных точек
        plt.scatter(
            X[:, 0], X[:, 1], c=Y, zorder=10, s=point_size, alpha=0.7,
            cmap=plt.cm.spring, edgecolor='black', linewidths=1.5
        )

        # Определение границ графика
        x_min = X[:, 0].min() - 1.5
        x_max = X[:, 0].max() + 1.5
        y_min = X[:, 1].min() - 1.5
        y_max = X[:, 1].max() + 1.5

        # Сетка точек в пространстве
        XX, YY = np.mgrid[x_min:x_max:500j, y_min:y_max:500j]
        # Значения решающей функции для этой сетки.
        # Чтобы их получить, нужно передать матрицу размера (N, 2)
        Z = model.decision_function(np.c_[XX.ravel(), YY.ravel()])
        # Ответы -- вектор. Переводим их к размеру сетки
        Z = Z.reshape(XX.shape)

        # Отрисовка решающей функции
        if margin:
            plt.pcolormesh(XX, YY, Z, cmap=plt.cm.RdBu)
        else:
            plt.pcolormesh(XX, YY, Z > 0, cmap=plt.cm.spring)

        # Отрисовка разделяющей прямой и разделяющей полосы
        plt.contour(
```

```
            XX, YY, Z, colors=['k', 'k', 'k'],
            linestyles=['--', '-', '--'],
            levels=[-1, 0, 1]
        )

        plt.xlim(x_min, x_max)
        plt.ylim(y_min, y_max)
        plt.xticks(())
        plt.yticks(())

        # Вычисление качества
        score = accuracy_score(Y_test, model.predict(X_test))
        # Значения гиперпараметров
        params_line = ', '.join([
            name + '=' + str(model.get_params()[name]) for name in params
        ])

        plt.title(params_line + ', {}, Acc. = {:.1f}%'.format(
            model.get_params()['kernel'], score*100
        ))

    plt.tight_layout()
    plt.show()
```

Разбейте выборки на обучающую и тестовую в соотношении 7:3.

Далее для каждой выборки и для каждого случая постройте сетку графиков:

- Линейное ядро, три значения `C`, сетка графиков 1x3
- Полиномиальное ядро степени 2, три значения `C` и три значения `gamma`, сетка графиков 3x3
- RBF-ядро степени 2, три значения `C` и три значения `gamma`, сетка графиков 3x3
- RBF-ядро степени 2, три значения `C` и три значения `gamma`, сетка графиков 3x3, визуализировать саму решающую функцию

Значения гиперпараметров подберите так, чтобы визуально получились "хороший", "средний" и "плохой" результаты. Какие значения гиперпараметров приводят к хорошему качеству визуально? По значению метрики? Какая форма разделяющей поверхности возможна для квадратичного ядра? Сделайте выводы.

Задача 5.

Оптимизационная задача в методе SVM-регрессии имеет вид.

$$\begin{cases} \frac{1}{2}\|\theta\|^2 + C\sum_{i=1}^{n}\left(\xi_i^+ + \eta_i^+\right) \longrightarrow \max\limits_{\theta,\theta_0,\xi,\eta} \\ Y_i - \varepsilon - \eta_i \leqslant \theta^T X_i + \theta_0 \leqslant Y_i + \varepsilon + \xi_i, \quad i = 1,\ldots,n, \end{cases}$$

где $\xi_i^+ = \max\{0,\ \xi_i\}$ и $\eta_i^+ = \max\{0,\ \eta_i\}$.

Сведите задачу к двойственной. Какие объекты являются опорными?

Задача 6.

Рассмотрим данные (https://archive.ics.uci.edu/ml/datasets/Forest+Fires) о лесных пожарах. Необходимо по различным показателям предсказать площадь лесного пожара. Описание данных приведено по ссылке.

1. Загрузите данные и постройте гистограмму целевого признака. Что можно сказать по гистограмме? Какое преобразование лучше совершить над ним?

In [ ]:

```
1
```

Сделайте это преобразование и еще раз посмотрите на гистограмму. Стало ли лучше?

In [ ]:

```
1
```

2. Оставьте только 4 признака: `'temp', 'RH', 'wind', 'rain'` , которые отвечают за температуру воздуха, влажность, скорость ветра и количество осадков соответственно.

In [ ]:

```
1
```

Разбейте выборку на обучающую и тестовую.

In [ ]:

```
1
```

Обучите линейную SVM-регрессию с $\varepsilon = 1$, предварительно отмасштабировав признаки.

In [ ]:

```
1
```

Какова доля опорных векторов? Визуализируйте гистограмму значений двойственных коэффициентов. Что можно по ней сказать?

In [ ]:

```
1
```

Посчитайте ошибки предсказаний для каждого объекта обучающей выборки. Какие значения ошибок имеют опорные векторы? Остальные?

In [ ]:

```
1
```

3. Подберите оптимальные гиперпараметры SVM: величины `C` , `epsilon` , ядро, а также параметры ядра. Перед обучением необходимо провести стандартизацию признаков.

Для реализации используйте `Pipeline` . Учтите, что в таком случае для сетки гиперпараметров их имена должны иметь вид `model__param` , где `model` -- имя модели в `Pipeline` , а `param` -- имя ее гиперпараметра.

In [ ]:

```
1
```

Посчитайте MAE-ошибку предсказания и доверительный интервал для нее. Перед этим не забудьте сделать обратное преобразование.

In [ ]:

```
1
```

4. Выполните п.3 для всех признаков и сравните качество полученных моделей. Является ли разница статистически значимой?

In [ ]:

```
1
```