Нормализация признаков

Данное задание основано на материалах лекции по линейным методам классификации.

Вы научитесь:

- работать с персептроном — простейшим вариантом линейного классификатора
- повышать качество линейной модели путем нормализации признаков

Введение

Линейные алгоритмы — распространенный класс моделей, которые отличается своей простотой и скоростью работы. Их можно обучать за разумное время на очень больших объемах данных, и при этом они могут работать с любыми типами признаков — вещественными, категориальными, разреженными. В этом задании мы предлагаем вам воспользоваться персептроном — одним из простейших вариантов линейных моделей.

Как и в случае с метрическими методами, качество линейных алгоритмов зависит от некоторых свойств данных. В частности, признаки должны быть нормализованы, то есть иметь одинаковый масштаб. Если это не так, и масштаб одного признака сильно превосходит масштаб других, то качество может резко упасть.

Один из способов нормализации заключается в стандартизации признаков. Для этого берется набор значений признака на всех объектах, вычисляется их среднее значение и стандартное отклонение. После этого из всех значений признака вычитается среднее, и затем полученная разность делится на стандартное отклонение.

Реализация в Scikit-Learn

В библиотеке scikit-learn линейные методы реализованы в пакете sklearn.linear_model. Мы будем работать с реализацией персептрона sklearn.linear_model.Perceptron. Как и у большинства моделей, обучение производится с помощью функции fit, построение прогнозов — с помощью функции predict.

Пример использования:

```
import numpy as np
from sklearn.linear_model import Perceptron
X = np.array([[1, 2], [3, 4], [5, 6]])
y = np.array([0, 1, 0])
clf = Perceptron()
clf.fit(X, y)
predictions = clf.predict(X)
```

В качестве метрики качества мы будем использовать долю верных ответов (accuracy). Для ее подсчета можно воспользоваться функцией sklearn.metrics.accuracy_score, первым аргументом которой является вектор правильных ответов, а вторым — вектор ответов алгоритма.

Для стандартизации признаков удобно воспользоваться классом sklearn.preprocessing.StandardScaler. Функция fit_transform данного класса находит параметры нормализации (средние и дисперсии каждого признака) по выборке, и сразу же делает нормализацию выборки с использованием этих параметров. Функция transform делает нормализацию на основе уже найденных параметров.

Пример использования:

```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
X_train = np.array([[100.0, 2.0], [50.0, 4.0], [70.0, 6.0]])
X_test = np.array([[90.0, 1], [40.0, 3], [60.0, 4]])
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
```

Инструкция по выполнению

1. Загрузите обучающую и тестовую выборки из файлов perceptron-train.csv и perceptron-test.csv. Целевая переменная записана в пер-

вом столбце, признаки — во втором и третьем.

2. Обучите персептрон со стандартными параметрами и random_state=241.
3. Подсчитайте качество (долю правильно классифицированных объектов, accuracy) полученного классификатора на тестовой выборке.
4. Нормализуйте обучающую и тестовую выборку с помощью класса StandardScaler.
5. Обучите персептрон на новых выборках. Найдите долю правильных ответов на тестовой выборке.
6. Найдите разность между качеством на тестовой выборке после нормализации и качеством до нее. Это число и будет ответом на задание.

Если ответом является нецелое число, то целую и дробную часть необходимо разграничивать точкой, например, 0.421. При необходимости округляйте дробную часть до трех знаков.

Ответ на каждое задание — текстовый файл, содержащий ответ в первой строчке. Обратите внимание, что отправляемые файлы не должны содержать перевод строки в конце. Данный нюанс является ограничением платформы Coursera. Мы работаем над тем, чтобы убрать это ограничение.