Московский авиационный институт
(Национальный исследовательский университет)
Институт информационных технологий и прикладной математики
Кафедра вычислительной математики и программирования

Лабораторная работа № 2

по курсу «Искусственный интеллект»

на тему «Алгоритмы классификации»

Студент: Шухова А.В.
Группа: М8О-307Б-17
Преподаватель: Самир Халид
Дата:
Оценка:
Подпись:

Москва, 2020

Постановка задачи:

Необходимо реализовать алгоритмы машинного обучения. Применить данные алгоритмы на наборы данных, подготовленных в первой лабораторной работе. Провести анализ полученных моделей, вычислить метрики классификатора. Произвести тюнинг параметров в случае необходимости. Сравнить полученные результаты с моделями реализованными в scikit-learn. Аналогично построить метрики классификации. Показать, что полученные модели не переобучились. Также необходимо сделать выводы о применимости данных моделей к вашей задаче.

- Логистическая регрессия
- KNN
- SVM
- Дерево Решений

Реализация

Логистическая Регрессия

Описание модели

Для большого класса распределений апостериорные вероятности задаются преобразованием softmax линейных функций переменных признаков $w_k^T \phi$. Используется максимальное правдоподобие для отдельного определения условных плотностей и априорных значений класса, а затем находятся соответствующие апостериорные вероятности с помощью теоремы Байеса. После использование максимальной вероятности для определения параметров этой модели напрямую, для этого требуются производные по всем активациям. Затем нужно записать функцию правдоподобия. Функция кросс-энтропийной ошибки для задачи мультиклассовой классификации: $E(w_1, \ldots, w_k) = -lnp(T|w_1, \ldots, w_k) = -\sum_{n=1}^{N}\sum_{k=1}^{K} t_{nk} lny_{nk}$

Результат выполнения

Реализованная модель.

Без регуляризации:

Ошибки на кросс валидации:

Accuracy = 0.4045112781954887

Precision = 0.4045112781954887

Roc auc = 0.6272036634337573

Ошибки на выборках:

	Train	Test
Accuracy	= 0.5	0.4992412746585736
Precision	= 0.5	0.4992412746585736
Roc auc	= 0.6720754157429131	0.672596230890465

регуляризация:

Оцененный параметр $\lambda = 1$

Ошибки на кросс валидации:
Accuracy = 0.3568922305764411
Precision = 0.3568922305764411
Roc auc = 0.579102713717152

Ошибки на выборках

Train	Test
Accuracy = 0.47380239520958084	0.4628224582701062
Precision = 0.47380239520958084	0.4628224582701062
Roc auc = 0.6019687936145997	0.6233187374443899

Логистическая регрессия с Sklearn.

Ошибка на выборках: Accuracy = 0.622154779969651

Ошибки на кросс валидации: Accuracy = 0.6889226100151745

Ошибки на выборках

Train	Test
Accuracy = 0.7155688622754491	0.6889226100151745
Precision = 0.7155688622754491	0.6889226100151745
Roc auc = 0.8097067621777602	0.7940238612896533

Выводы:

1. Метрики классификации на train и test выборках не отличаются, ⇒ модель не переобучилась.

2. Модель логистической мультиклассовой регрессии заняла довольно много времени и показалась непростой. Из полученных оценок можно сделать вывод, что данная модель не подходит для решения задач мультиклассовой классификации.

KNN

Описание модели

Алгоритм

Операции последовательно выполняются для классификации каждого из объектов тестовой выборки: вычисление расстояние до каждого из объектов обучающей выборки, отбор k объектов обучающей выборки, расстояние до которых минимально, класс классифицируемого объекта — это класс, наиболее часто встречающийся среди k ближайших соседей

Выбрано двумерное пространство, в котором случайным образом на участке от 0 до 5 по каждой из осей выбирается местоположение мат. ожидания со среднеквадратичным

отклонением 0.5. Для определения расстояния между объектами можно использовать евклидово расстояние, косинусная мера, критерий корреляции Пирсона и др.
Основная функция алгоритма: на входе матрица расстояний между объектами обучающей и тестовой выборки, число ближайших «соседей», на выходе предсказанные метки для новых объектов и вероятности каждой метки.

Результат выполнения

Реализованная модель

k — ближайших соседей = 5

CV scores: [0.8951310861423221, 0.9026217228464419, 0.9213483146067416, 0.9213483146067416, 0.947565543071161]

Train data accuracy: 0.9176029962546817

Ошибки на кросс валидации:
Accuracy = 0.9241274658573596
Precision = 0.9241274658573596
Roc auc = 0.9495108095293339

Ошибки на выборках:

	Train	Test
Accuracy =	0.9520958083832335	0.9241274658573596
Precision =	0.9520958083832335	0.9241274658573596
Roc auc =	0.9680700989877046	0.9495108095293339

KNN с Sklearn.

k — ближайших соседей = 5

Ошибка на выборках: Accuracy = 0.9241274658573596

Ошибки на выборках:

	Train	Test
Accuracy =	0.9520958083832335	0.9241274658573596
Precision =	0.9520958083832335	0.9241274658573596
Roc auc =	0.9680700989877046	0.9495108095293339

Выводы:

1. Метрики классификации на train и test выборках не отличаются, ⇒ модель не переобучилась.

2. Несмотря на простоту, модель KNN на train и test данных выдает хороший результат — accuracy свыше 0.9.

SVM

Описание модели

Мультиклассовые SVM классифицируют входной вектор x в один из k классов, используя правило: $\hat{y} = \underset{m \in |k|}{argmax}\, \omega_m^T$**.** Каждый вектор ω_n можно рассматривать как прототип, представляющий m-й класс, а ω^T - как оценку m-го класса по x. Следовательно, уравнение выбирает класс с наивысшим баллом. Для n обучающих примеров $x_i \in R^n$ и связанных с ними меток $y_i \in$ [k] мультиклассовая SVM-формулировка оценивает оценки $\omega_1, \ldots, \omega_n$, решив задачу оптимизации. Градиент f определяется как $g_i^m = \frac{\partial f}{\partial \alpha_i^m} = \omega_m(\alpha)^T x_i + \Delta_i^m, \forall i \in [n], \forall m \in [k]$. В таком случае, оптимальное решение: $v_i = \underset{m \in [k]}{max}\, g_i^m - \underset{m \in [k]: \alpha_i^m < C_i^m}{min} g_i^m, \forall i \in [n]$.

Результат выполнения

Реализованная модель.

C = 2

Ошибки на кросс валидации:
Accuracy = 0.5548872180451128
Precision = 0.5548872180451128
Roc auc = 0.6726917825855202

Ошибки на выборках:

	Train	Test
Accuracy =	0.4588323353293413	0.44764795144157815
Precision =	0.4588323353293413	0.44764795144157815
Roc auc =	0.6191507423349809	0.6112348013912755

SVM с Sklearn.

C = 21

Ошибки на выборках:

	Train	Test
Accuracy =	0.9468562874251497	0.9559939301972686
Precision =	0.9468562874251497	0.9559939301972686
Roc auc =	0.9645010781616375	0.9707807420761585

Выводы:

1. Метрики классификации на train и test выборках не отличаются, ⇒ модель не переобучилась.

2. Модель показала accuracy 0.45, а sklearn 0.94. Параметр был выбран не правильно, так как поиск происходил очень долго, поэтому accuracy на модели низкая.

Дерево решений

Описание модели

Дерево решений — это двоичное дерево, узел которого одна входная переменная (x) и точка ее разделения, а конечные узлы дерева содержат выходную переменную (y), которая используется для прогнозирования.

Алгоритм

1. На каждом шаге выбирается признак, при разделении по которому прирост информации оказывается наибольшим.
2. Выборка делится на левую и правую часть: выбирается разделение с лучшей стоимостью и все возможные точки разделения оцениваются и выбираются наилучшие.
3. Процедура повторяется рекурсивно к каждой из частей, пока энтропия не окажется очень малой величине или равной нулю.

Результат выполнения

Реализованная модель.

Максимальная глубина = 9

Минимальная запись узла = 9

Ошибки на кросс валидации:
Accuracy = 0.8441102756892229
Precision = 0.8441102756892229
Roc auc = 0.8965187796097869

Ошибки на выборках:

	Train	Test
Accuracy =	0.9461077844311377	0.8194233687405159
Precision =	0.9461077844311377	0.8194233687405159
Roc auc =	0.9641322657183546	0.8795584678359948

Дерево решений с Sklearn.

Максимальная глубина = 6

Минимальная запись узла = 9

Ошибки на выборках:

	Train	Test
Accuracy =	0.9558383233532934	0.8270106221547799
Precision =	0.9558383233532934	0.8270106221547799
Roc auc =	0.9705206250064274	0.8851529675356118

Выводы:

1. Метрики классификации на train и test выборках не отличаются, ⇒ модель не переобучилась.

2. Из полученных результатов видно, что дерево решений довольно хорошо подходит для решения данной задачи.