Московский авиационный институт (национальный исследовательский университет)

Факультет информационных технологий и прикладной математики

Кафедра вычислительной математики и программирования

Лабораторная работа №2 по курсу «Искусственный интеллект»

Студент: А. С. Усков
Преподаватель: А. С. Халид
Группа: М8О-306Б

Дата:
Оценка:
Подпись:

Москва, 2020

Московский авиационный институт (национальный исследовательский университет)

Факультет информационных технологий и прикладной математики

Кафедра вычислительной математики и программирования

Лабораторная работа №2 по курсу «Искусственный интеллект»

Студент: А. С. Усков
Преподаватель: А. С. Халид
Группа: М8О-306Б

Дата:
Оценка:
Подпись:

Москва, 2020

Лабораторная работа №2

Задача:

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.

1. логистическая регрессия
2. KNN
3. SVM
4. дерево решений
5. random forest

1 Описание

Для демонстрации работы моделей использовался первый датасет из первой лабораторной работы. Сам датасет был разделён случайным образом на обучающий и тестирующий сеты в соотношении 4:1, и полученные сеты использовались для всех моделей, кроме моей реализации KNN для неё сеты дополнительно были посемплированы в 100 раз, для уменьшения времени работы.

На тестирующих выборках считались 4 метрики: accuracy, precision, f1_score и recall. По метрикам можно понять следующее:

1. Модель логистической регрессии неприменима для данного датасета - это видно по тому, что и моя и sklearn модели дают одинаковый результат, а при более подробном рассмотрении результатов оптимизации видно, что минимизируемая функция уходит в бесконечность - это объясняет совпадение метрик.

2. Модель KNN даёт приемлемую точность, хотя моя реализация незначительно проигрывает sklearn'y.

3. Моя модель SVM определённо испытывает трудности с оптимизацией лосса, что обусловлено отсутствием "умного"подбора параметров оптимизации. SVM из sklearn обучается, но предупреждает, что итераций обучения слишком мало - их 1000 для сходимости обучения, без ограничения на количество итераций обучение шло слишком долго, и я полагаю, что потенциально бесконечно, а точность на тестирующей выборке практически не растёт. Такие результаты по-разному, но показывают, что исходные данные не разделимы линейно.

4. Решающие деревья переобучаются, что является полностью ожидаемым поведением, из результатов можно видеть, что в обучающем и тестирующем датасете есть очень похожие записи.

5. Случайный лес стабильно даёт хорошие результаты - лучшие из всех моделей. При этом моя реализация ожидаемо проигрывает sklearn'овской.

Далее я подробнее опишу мою реализацию каждой из моделей:

1. Логистическая регрaессия - создаётся callable классы, соответствующие функции похожести, якобиану функции похожести и гессиану функции похожести, эти три функции передаются в оптимизатор, для которого дополнительно задаётся ограничение максимального количества итераций, равного введённому количеству эпох. Оптимизатор вызывается из модуля scipy в режиме trust-ncg - Ньютоновского метода сопряжённых векторов с доверительными интервалами.

2. KNN - на основе обучающей выборки строится k-d-дерево, при применении вместо классического поиска одного соседа запоминаются не больше k ближайших.

3. Для SVM создаются callable классы, соответствующие функции лосса и якобиану функции лосса, эти функции передаются в оптимизатор, работающий в режиме CG - метода сопряжённых градиентов следующим образом: в течение каждой эпохи оптимизация вызывается с параметром maxiter = 1 столько раз, сколько точек в обучающей выборке, это повторяется столько раз, сколько задано эпох.

4. Решающее дерево я в итоге решил строить методом дихотомии: для каждого фактора методом максимального прироста информации находится точка разделения на 2 множества, из всех факторов находится тот, который даёт наибольший прирост.

5. Случайный лес - для каждого дерева случайно с повторениями выбираются n точек, из них случайно выбирают m факторов, по умолчанию m - корень количества факторов. В качестве предсказания берётся тот класс, за который больше всего голосов.

Вообще говоря, задача датасета - прогнозирование погоды, по-хорошему данные следовало разделить по временной метке и тестироваться на отложенном во времени периоде. Вероятно из-за того, что я этого не сделал я получил небывало высокую точность для решающих деревьев - размазанные по всему периоду данные имеют повторяющиеся или чрезвычайно похожие записи.