Задание

Решаем задачу бинарной классификации: то есть целевой признак имеет два значения, согласно традиции: «+» и «-». Есть обучающая выборка (train dataset), для которой известны метки классов.

Есть тестовая выборка (test dataset), для которой по значениям других признаков надо восстановить значение класса.

Кроме того, в первой постановке предполагается, что признаки также имеют бинарную природу.

Используем метод ленивой классификации, то есть не строим некоторую функцию решающего правила на основе обучающей выборки, а принимаем решения для конкретного объекта, как только он поступает.

Для решения задачи предлагается использовать следующий метод на основе «генераторов»

Метод на основе «генераторов»

Контексты для плюс- и минус-примеров рассматриваются по отдельности.

Для классифицируемого объекта g следует выполнить:

- Для каждого объекта из плюс-контекста рассмотреть пересечение его описания (g_i^+) и описания классифицируемого (g'), и проверить, вкладывается ли оно в описание какого-либоиз минус-примеров (g_i^-)

- Для каждого объекта из минус-контекста рассмотреть пересечение его описания (g_i^-) и описания классифицируемого (g'), и проверить, вкладывается ли оно в описание какого=либо из плюс-примеров (g_i^+)

Необходимо исследовать ввозможные пороговые функции классификации (например, классифицировать объект соответствующим образом, если для пересечения поддержка больше порогового значения min_supp), разрешающие некторую долю ошибки (параметр) на обучающей выборке.

Здесь и далее используются следующие обозначения:

$|.|$ – мощность

g' – описание классифицируемого примера

g^+ – вычисление замыкания в плюс контексте

g_i^+, g_i^- —описание положительного и отрицательного примера, соответственно

Примеры вычислимых характеристик, которые могут участвовать в итоговом правиле классификации:

$|g' \cap g_i^+|$ – мощность пересечения

$|(g' \cap g_i^+)^+|$ – поддержка (релизована в прилагаемом скрипте)

$|(g' \cap g_i^+)^-|$ – достоверность

Пример аггрегатной функции:

$$Aggr_i|g' \cap g_i^+|: \sum_{i \in i^+} |(g' \cap g_i^+)^+|$$

Для Вашего удобства в файлах train#.csv, test#.csv помещены обучающие и тестовые данные (по массиву Tic-Tac-Toe из репозитория UCI Machine Learning Repository). Приветствуется использование других массиово данных из репозитория UCI Machine Learning Repository. Необходимо подобрать параметры модели ленивого обучения, дающие наибольшую среднюю (по всем #) точность по скользящему контролю (кросс-валидации) на тестовых выборках.

В файле svn/trunk/generator/test_generators.py приводится реализация простейшего правила классификации: если $g' \cap g_i^+$ не вкладывается ни в одно содержание отрицательного примера, то объект классифицируется положительно. Эту программу можно подстроить под другие правила классификации, лучшие из которых Вам предстоит найти.

Все манипуляции с кодом предлагается проводить в рамках системы контроля версий svn

SVN CHECKOUT ***HTTPS***://LAZY-LEARNING-FCA.GOOGLECODE.COM/SVN/TRUNK/ LAZY-LEARNING-FCA –USERNAME login@gmail.com

Предлагается создать там личную папку, например, копированием папки trunk/generator и вести работу в своей ветке

Краткая инструкция по svn:

http://tortoisesvn.net - Хороший клиент для систем windows. Встраивается в выпадающее меню

Для Unix/Linux/MacOS достаточно консольной версии

Ссылка на полное руководство к действию для работы с консольным - http://doc.hive.kiev.ua/svn/svn.ref.html

Основные команды:

Svn checkout – скопировать данные из репозитория на ваш локальный компьютер (при нормальной работе делается однократно)

Svn up – обновить данные в рабочей папке, согласно последним данным в репозитории (нужно делать регулярно)

Svn add - сказать системе контроля версий учитывать ваш файл/директорию как подконтрольную единицу (хранить историю изменений)

Svn commit – отправить данные с локальной машины в репозиторий

Предлагается следующий сценарий работы:

Вы делаете

SVN CHECKOUT ***HTTPS***://LAZY-LEARNING-FCA.GOOGLECODE.COM/SVN/TRUNK/ LAZY-LEARNING-FCA –USERNAME login@gmail.com

После чего в полученной папке создаете поддиректорию <username> где храните все ваши наработки по коду

Сразу имеет смысл добавить всю папку под версионирование командой svn add. Внутри папки вы можете создавать поддиректории и т.д. Базовые скрипты рекомендуется копировать себе из папки lazy-learning-fca/generator/

Структура репозитория:

Description.pdf – описание задачи (этот файл)

Data_sets – директория с наборами данных

Cross_validation – директория со скриптами необходимыми для проведения кросс-валидации (пока подготовка данных)

Generator – базовая реализация классификации по «генераторам» для набора данных Tic-Tac-Toe

Implication – базовая реализация классификации на основе импликаций