Логистическая регрессия

Введение в машинное обучение

Введение

О, нет! С момента своего создания знаменитая школа волшебников Хогвартс никогда не знала подобного преступления. Силы зла заколдовали Распределяющую шляпу. Она больше не отвечает и неспособна выполнять свою роль по распределению учеников по факультетам. Приближается новый учебный год. К счастью, профессор МакГонагалл смогла принять меры в такой стрессовой ситуации. Она решила обратиться к вам, магловскому «специалисту по данным», способному творить чудеса с помощью инструмента, которым все маглы умеют пользоваться – компьютером. Несмотря на внутреннее нежелание многих волшебников, директор школы приглашает вас в свой кабинет, чтобы объяснить ситуацию. Вы здесь, потому что его информатор обнаружил, что вы можете воссоздать волшебную Распределяющую шляпу, используя свои магловские инструменты. Вы объясняете, что для того, чтобы ваши магловские инструменты работали, вам нужны данные студентов. Профессор МакГонагалл нерешительно дает вам пыльную книгу заклинаний. К счастью для вас, книга мгновенно превратилась в архив данных.

Цель проекта

В этом проекте «Логистическая регрессия» вы продолжите свое изучение машинного обучения, открывая для себя новые инструменты. Мы познакомимся только с некоторыми основами, которые будут полезными для исследования данных перед отправкой их в алгоритм машинного обучения.

Вы реализуете линейный классификатор на базе такого алгоритма машинного обучения, как логистическая регрессия. Таким образом,

- Вы научитесь читать наборы данных, визуализировать их разными способами, выбирать только действительно значимые и очищать их.
- Вы обучите модель логистической регрессии, которая решит задачу классификации.

Основные инструкции

Язык программирования – Python. Вы также можете использовать любые библиотеки, которые захотите, но только, если они не делают всю работу за вас. Например, использование функции `describe` библиотеки Pandas или `linear_model.LogisticRegression (настройка multi_class=”ovr”)` из Scikit-learn будут считаться читерством.

Основная задача

1. Анализ данных

Прежде всего, взгляните на имеющиеся данные. Посмотрите, в каком формате они представлены, есть ли различные типы данных, каковы их диапазоны и т.д. Важно составить представление о вашем исходном материале, прежде чем приступить к построению модели. Чем больше вы работаете с данными, тем больше у вас развивается интуиция о том, как вы сможете их использовать. В этом пункте профессор МакГонагалл просит вас создать программу под названием `describe.py`. Эта программа будет принимать

набор данных в качестве параметра. Все, что ей нужно сделать, это отобразить информацию для всех числовых характеристик, как в примере:

```
$> describe.[extension] dataset_train.csv
          Feature 1     Feature 2     Feature 3     Feature 4
Count    149.000000    149.000000    149.000000    149.000000
Mean       5.848322      3.051007      3.774497      1.205369
Std        5.906338      3.081445      4.162021      1.424286
Min        4.300000      2.000000      1.000000      0.100000
25%        5.100000      2.800000      1.600000      0.300000
50%        5.800000      3.000000      4.400000      1.300000
75%        6.400000      3.300000      5.100000      1.800000
Max        7.900000      4.400000      6.900000      2.500000
```

Запрещено использовать любую функцию, которая делает работу за вас, например, count, mean, std, min, max, percentile и т. д.

2. Визуализация данных

Визуализация данных – мощный инструмент для специалиста по данным. Она позволяет вам делать выводы и развивать интуицию относительно того, как выглядят ваши данные. Визуализация данных также позволяет вам обнаруживать дефекты или аномалии (выбросы).

В этом разделе вам предлагается создать набор скриптов, каждый из которых использует определенный метод визуализации для ответа на вопрос. На вопрос не обязательно существует один ответ.

2.1. Гистограмма

Создайте скрипт с именем histogram.py, который отображает гистограммы, отвечающие на следующий вопрос:

Какой курс Хогвартса имеет равномерное распределение баллов между всеми четырьмя факультетами?

2.2. Диаграмма рассеяния

Создайте скрипт с именем scatter_plot.py, который отображает диаграммы рассеяния, отвечающие на следующий вопрос:

Какие два признака похожи?

2.3. Парная диаграмма

Создайте сценарий с именем pair_plot.py, который отображает парную диаграмму или матрицу диаграммы рассеяния (в зависимости от используемой вами библиотеки).

Какие признаки, полученные в результате этой визуализации, вы собираетесь использовать для логистической регрессии?

3. Логистическая регрессия

Создайте свою Распределяющую шляпу. Для этого вам нужно реализовать классификатор на базе логистической регрессии с использованием стратегии «один против всех» (one vs all). См., например, https://proproprogs.ru/ml/ml-mnogoklassovaya-klassifikaciya-metody-one-vs-all-i-all-vs-all.

Вам нужно создать две программы:

- Первая – для обучения модели, она называется lg_train.py. Она принимает в качестве параметра dataset_train.csv. Для обучения вы должны использовать технику градиентного спуска, чтобы минимизировать ошибку. Программа генерирует файл, содержащий веса, которые будут использоваться для прогнозирования.
- Вторая должна называться lg_predict.py. Она принимает в качестве параметра dataset_test.csv и файл, содержащий веса, обученные предыдущей программой.

Чтобы оценить качество вашего классификатора, вторая программа должна будет генерировать файл прогнозирования houses.csv, отформатированный точно так же, как показано ниже:

```
$> cat houses.csv
Index,Hogwarts House
0,Gryffindor
1,Hufflepuff
2,Ravenclaw
3,Hufflepuff
4,Slytherin
5,Ravenclaw
6,Hufflepuff
```

Ваш классификатор будет оценен на основе данных, представленных в dataset_test.csv. Ваши ответы будут оценены с использованием оценки точности библиотеки Scikit-learn. Профессор МакГонагалл соглашается, что ваш алгоритм сопоставим с Распределяющей шляпой, только если его минимальная точность составляет 98%. Также важно объяснить в отчете работу используемых алгоритмов машинного обучения.

4. Стохастический градиентный спуск

Реализуйте стохастический градиентный спуск. Сравните результаты работы программ с разными версиями спуска. Какие выводы можно сделать?

Сдача проекта

Рабочие программы с исходным кодом должны быть размещены в вашей индивидуальной папке на Яндекс Диске (под студенческой учетной записью) Основы Python_Фамилия\Линейная регрессия. В этой же папке необходимо разместить отчет, оформленный средствами LaTex или Word с описанием вашей работы (необходимо включить формулы, код, выкладки и рассуждения, которыми вы пользовались при реализации проекта).

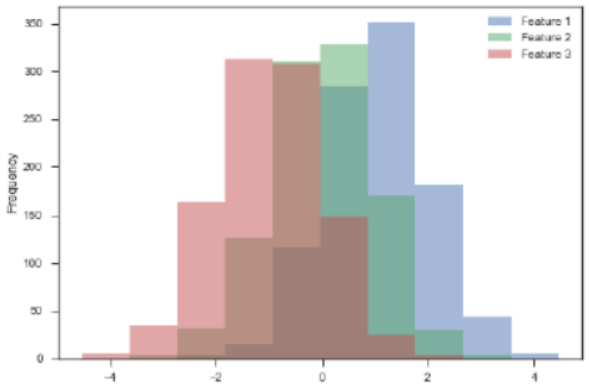
Необходимо предоставить доступ к указанной папке Основы Python_Фамилия преподавателю.

Время на выполнение – 2 недели. Последний срок размещения проекта на Яндекс Диске **04.12.2024**. Проверка проекта осуществляется очно на занятии.

ПРИЛОЖЕНИЕ

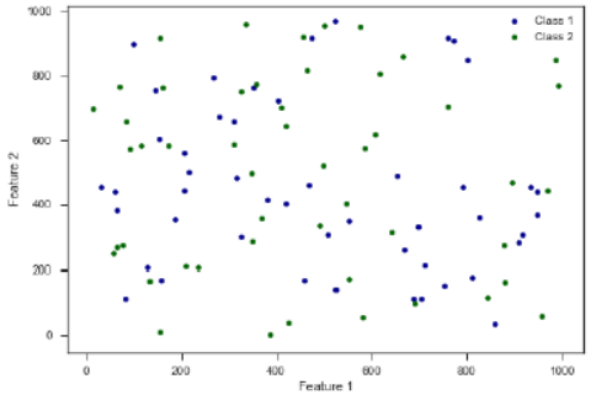
Примеры визуализации данных.

- Гистограмма

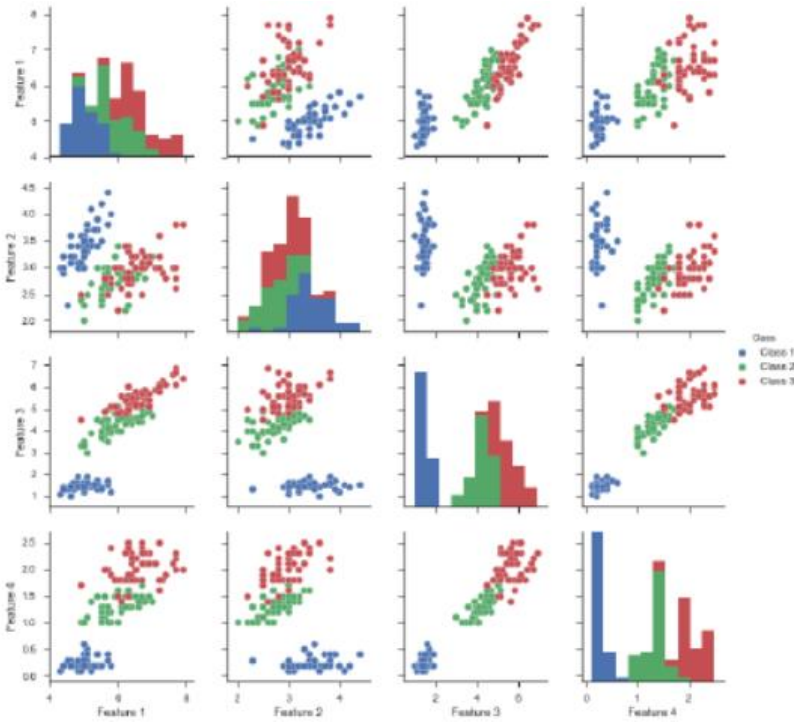


- Диаграмма рассеяния

Логистическая регрессия. Линейный классификатор



- Парная диаграмма



Математика.

Логистическая регрессия работает почти так же, как линейная регрессия. Вот loss функция (или функция потерь):

$$J(\theta) = -\frac{1}{m}\sum_{i=1}^{m} y^i \log\left(h_\theta\left(x^i\right)\right) - (1-y^i)\log\left(1-h_\theta\left(x^i\right)\right),$$

где функция

$$h_\theta(x) = g(\theta^T x)$$

и

$$g(z) = \frac{1}{1+e^{-z}}$$

Частные производные могут быть вычислены по формулам:

$$\frac{\partial}{\partial \theta_j} J(\theta) = \frac{1}{m} \sum_{i=1}^{m} \left(h_\theta \left(x^i \right) - y^i \right) x_j^i$$