Работа в Pandas на примере задачи «Titanic»

Цели практики:

• работать с данными используя язык Python и пакет Pandas

• делать предобработку данных

• находить простые закономерности в данных

•обучать решающие деревья

•находить наиболее важные для них признаки

Начало работы

Для того, чтобы начать работать с данными, необходимо сначала загрузить их из файла. В данном задании мы будем работать с данными в формате CSV, предназначенном для хранения табличных данных: столбцы разделяются запятой, первая строка содержит имена столбцов.

Задание №1

Загрузите датасет titanic.csv и, используя описанные выше способы работы с данными, найдите ответы на вопросы:

1. Какое количество мужчин и женщин ехало на корабле? В качестве ответа приведите два числа через пробел.
2. Какой части пассажиров удалось выжить? Посчитайте долю выживших пассажиров. Ответ приведите в процентах (число в интервале от 0 до 100, знак процента не нужен), округлив до двух знаков.
3. Какую долю пассажиры первого класса составляли среди всех пассажиров? Ответ приведите в процентах (число в интервале от 0 до 100, знак процента не нужен), округлив до двух знаков.
4. Какого возраста были пассажиры? Посчитайте среднее и медиану возраста пассажиров. Посчитайте среднее и медиану возраста пассажиров. В качестве ответа приведите два числа через пробел.
5. Коррелируют ли число братьев/сестер с числом родителей/детей? Посчитайте корреляцию Пирсона между признаками SibSp и Parch.
6. Какое самое популярное женское имя на корабле? Извлеките из

полного имени пассажира (колонка Name) его личное имя (First Name), что является типичный пример того, с чем сталкивается специалист по анализу данных.

Ответ на каждое задание — текстовый файл, содержащий ответ в первой строчке (#.txt). Обратите внимание, что отправляемые файлы не должны содержать перевод строки в конце.

Утонения по выполнению задания:

Если ответом является нецелое число, то целую и дробную часть необходимо разграничивать точкой, например, 0.42. При необходимости округляйте дробную часть до двух знаков.

Данные очень разнородные и шумные, но из них требуется извлечь необходимую информацию, для этого попробуйте вручную разобрать несколько значений столбца Name и выработать правило для извлечения имен, а также разделения их на женские и мужские.

Рекомендуем в данном практикуме использовать библиотеки Pandas, NumPy и SciPy, которые существенно упрощают чтение, хранение и обработку данных.

Пример загрузки данных в Pandas:

import pandas

data = pandas . read_csv(' titanic . csv ' , index_col='PassengerId ').

Данные будут загружены в виде **DataFrame**, с помощью которого можно удобно работать с ними.

В данном случае параметр **index_col=' PassengerId** ' означает, что колонка **PassengerId** задает нумерацию строк данного датафрейма.

Для того, чтобы посмотреть, что представляют из себя данные, можно воспользоваться несколькими способами:

- более привычным с точки зрения Python (если индекс указывается только один, то производится выбор строк): **data [: 10]**
- воспользоваться методом датафрейма: **data . head ( )**

Один из способов доступа к столбцам датафрейма — использовать квадратные скобки и название столбца:

data [' Pclas s ']

Для подсчета некоторых статистик (количества, среднее, максимум, минимум) можно также использовать методы датафрейма:

data [' Pclass '] . value_counts ()

Более подробно со списком методов датафрейма можно познакомиться в документации.

Задание №2

В этом задании мы вновь рассмотрим данные о пассажирах Титаника с применением библиотеки scikit-learn.

Будем решать на них задачу классификации, в которой по различным характеристикам пассажиров требуется предсказать, кто из них выжил после крушения корабля.

Решающие деревья относятся к классу логических методов. Их основная идея состоит в объединении определенного количества простых решающих правил, благодаря чему итоговый алгоритм является интерпретируемым. Как следует из названия, решающее дерево представляет собой бинарное дерево, в котором каждой вершине сопоставлено некоторое правило вида **«j-й признак имеет значение меньше b».** В листьях этого дерева записаны числа-предсказания. Чтобы получить ответ, нужно стартовать из корня и делать переходы либо в левое, либо в правое поддерево в зависимости от того, выполняется правило из текущей вершины или нет.

Одна из особенностей решающих деревьев заключается в том, что они позволяют получать важности всех используемых признаков. Важность признака можно оценить на основе того, как сильно улучшился критерий качества благодаря использованию этого признака в вершинах дерева.

1. Загрузите выборку из файла titanic.csv с помощью пакета Pandas.
2. Оставьте в выборке четыре признака: класс пассажира **(Pclass),** цену билета **(Fare),** возраст пассажира **(Age)** и его пол **(Sex).**
3. Обратите внимание, что признак Sex имеет строковые значения.
4. Выделите целевую переменную — она записана в столбце **Survived.**
5. В данных есть пропущенные значения — например, для некоторых пассажиров неизвестен их возраст. Такие записи при чтении их в **pandas** принимают значение **nan.** Найдите все объекты, у которых есть пропущенные признаки, и удалите их из выборки.
6. Обучите решающее дерево с параметром **random_state=241** и остальными параметрами по умолчанию (речь идет о параметрах конструктора **DecisionTreeClassifier).**
7. Вычислите важности признаков и найдите два признака с наибольшей

важностью. Их названия будут ответами для данной задачи (в качестве ответа укажите названия признаков через запятую или пробел, порядок не важен).

Ответ на каждое задание — текстовый файл, содержащий ответ в первой строчке. Обратите внимание, что отправляемые файлы не должны содержать перевод строки в конце.

Утонения по выполнению задания:

В библиотеке scikit-learn решающие деревья реализованы в классах **sklearn.tree.DecisionTreeClassifier (для классификации)** и **sklearn.tree.DecisionTreeRegressor (для регрессии).**

Обучение модели производится с помощью **функции fit.**

Пример использования:

```
import numpy as np
from sklearn . t r e e import D e c i s i o n T r e e C l a s s i f i e r
X = np . array ( [ [ 1 , 2 ] , [ 3 , 4 ] , [ 5 , 6 ] ] )
y = np . array ( [ 0 , 1 , 0 ] ) c l f = D e c i s i o n T r e e C l a s s i f i e r ( )
c l f . f i t (X, y )
```

В этом задании вам также потребуется находить важность признаков.

Это можно сделать, имея уже обученный классификатор:

```
importances = c l f . feature_importances_
```

Переменная **importances** будет содержать массив «важностей» признаков.

Индекс в этом массиве соответствует индексу признака в данных.

Стоит обратить внимание, что данные могут содержать пропуски.

Pandas хранит такие значения как **nan (not a number).**

Для того, чтобы проверить, является ли число **nan'ом,** можно воспользоваться функцией **np.isnan.**

Пример использования:

```
np . isnan (X)
```