Домашняя работа №1. Дедлайн Четверг 29 Февраля 18:00.

В этой домашней работе вы практикуетесь с Hadoop Streaming, а именно - инференс модели на основе sklearn на кластере. При таком подходе модель обучается на (относительно) небольшом семпле, а инференс может осуществляться параллельно на кластере на датасете любого размера.

В этом задании требуется не просто обучить модель и сделать предсказания, а написать скрипты, для которых пути к датасетам и сама модель будут являться аргументами. Это позволяет легче выводить модели в прод.

Тьюториал

https://github.com/datamove/ai-masters-bigdata/tree/master/projects/tut1

Чтобы вам было проще начать работу, у нас приготовлен тьюториал, в котором реализовано все, что требуется, только для немного другой задачи и датасета. Разберите его и решайте вашу задачу используя скрипты из этого тьюториала.

Задача

За основу взят датасет и соревнование Criteo Display Advertising challenge
https://www.kaggle.com/c/criteo-display-ad-challenge/overview

Задание

Предскажите вероятность клика на рекламный баннер. Признаки включают в себя некоторое количество целочисленных и категориальных переменных. Данные в обучающем датасете покрывают 7 последовательных дней, данные валидационного датасета покрывают 8-ой день.

Первая колонка датасета - идентификатор записи - целое число.

Вторая - целевая переменная: 1 - клик, 0 - нет клика

Следующие 13 - целочисленные признаки, имеются пропуски.

Следующие 26 - категориальные признаки, так же имеются пропуски

Для простоты, возьмите следующее определение полей датасета:

```python

numeric_features = ["if"+str(i) for i in range(1,14)]

categorical_features = ["cf"+str(i) for i in range(1,27)]

fields = ["id", "label"] + numeric_features + categorical_features

```

Метрика - Log loss (http://wiki.fast.ai/index.php/Log_Loss).

Более подробное описание задачи можно найти на сайте соревнования по ссылке выше.

Задание очень похоже на примерное задание
https://github.com/datamove/ai-masters-bigdata/blob/master/projects/tut1/README.md

Однако просто скопировать весь код не удастся, надо будет немного поработать над ним.

Вы можете пользоваться моделями, описанными на форуме соревнования, искать среди них идеи, но ваше решение все равно должно укладываться в заданные рамки оформления работы, см. ниже. Мы не устраиваем соревнования, и поэтому вам не стоит прилагать лишних усилий по улучшению метрики.

Условие прохождения

Превышение baseline метрики (0.99) на тестовом датасете.

Dataset

Обратите внимание, что разделитель в этих датасетах - TAB. Сохраняйте этот разделитель на этапах фильтрации и предсказания.

Обучающий

Обучающий датасет расположен в локальной файловой системе на вашем login-узле:

`/home/users/datasets/criteo/train1000.txt`

Тестовый датасет для работы со срезами.

Это большой, на 11 Гб датасет, на котором можно применять условия фильтрации.

HDFS: /datasets/criteo/train-with-id.txt

Имейте в виду, что на тестовом датасете вырезана вторая колонка ('label').

Для отладки вы можете взять небольшой сэмпл этого датасета. Напишите map-reduce задачу для этого.

Условие среза

Условие для реализации в функции `filter_cond.py` (см. ниже) таково:

Значение в поле `if1` (первое числовое поле) таково, что `20 < if1 < 40`.

Оформление работы

Создайте в вашем существующем (после домашки №0) репозитории ai-masters-bigdata подпапку projects/1a.

Модель

Разработайте модель в виде пайплайна и сохраните ее определение в отдельном файле `projects/1a/model.py`. Объект пайплайна модели должен называться `model`, например:

```python

model = Pipeline(steps=[

        ('preprocessor', preprocessor),

        ('linearregression', LinearRegression())

])

```

и тогда ее можно будет импортировать в других программах как:

`from model import model`

Также в файле с моделью мы определяем поля датасета:

```python

numeric_features = ["if"+str(i) for i in range(1,14)]

categorical_features = ["cf"+str(i) for i in range(1,27)]

fields = ["id", "label"] + numeric_features + categorical_features

```

тогда их тоже можно будет импортировать подобным образом:

`from model import fields`

Training

Обучение модели проводится на обучающем датасете на одном узле. Разработайте программу train.py, которая импортирует ранее определенную модель и обучает её. На вход в качестве аргумента подается

- номер проекта
- путь к файлу с обучающей выборкой.

Обученная модель сохраняется в файл 1a.joblib используя сериализатор joblib:

```python

from joblib import dump

dump(model, "1a.joblib")

```

Запуск обучения

Напишите shell-wrapper для train.py, который будет запускаться следующим образом:

```bash
```

```
cd ai-masters-bigdata

projects/1a/train.sh 1a /path/to/training/dataset
```

где 1 - номер проекта, `/path/to/training/dataset` - путь к файлу с тренировочной выборкой.

Предсказания (инференс)

Напишите программу predict.py, которая загружает обученную модель и сохраненную ранее модель:

```python
from joblib import load

model = load("1a.joblib")
```

и выдает предсказания на стандартный вывод для тестовых записей, которые подаются на стандартный ввод.

Запуск инференса

Напишите shell-wrapper predict.sh для запуска инференса на валидационном датасете как map-reduce задачи:

```bash
cd ai-masters-bigdata

#remove output dataset if exists

hdfs dfs -rm -r -f -skipTrash predicted.csv

projects/1a/predict.sh projects/1a/predict.py,1a.joblib /datasets/criteo/train-with-id.txt predicted.csv predict.py
```

где параметры:

- файлы для посылки с задачей (включая тренированную модель)
- путь к валидационному датасету
- путь к файлу с предсказаниями
- скрипт для запуска

Фильтрация датасета

Разработайте скрипт `filter.py` для фильтрации датасета, который берет записи из датасета на стандартном входе, применяет некоторую функцию фильтрации и выводит записи, прошедшие фильтр, на стандартный вывод.

Фильтрующая функция определяется в файле `filter_cond.py` и имеет следующий интерфейс:

```python
def filter_cond(line_dict):
    """Filter function

    Takes a dict with field names and values as the argument

    Returns True if conditions are satisfied

    """

    cond_match = (

    int(line_dict["num_reviews"]) > 20

    )

    return True if cond_match else False
```

Запуск фильтрации

Разработайте `filter.sh`, который должен запускать map-reduce задачу на кластер:

Параметры скрипта `filter.sh`:

- файлы, которые надо послать вместе с задачей, через запятую
- путь к входному файлу
- путь к выходному файлу
- имя файла с программой маппером, то есть `filter.py`

Предсказания на срезе

Выше мы запускали отдельно фильтрацию и предсказания. Теперь мы запустим одну mapreduce задачу в которой мы будем фильтровать датасет на стадии map и предсказывать на стадии reduce. Преимущество - всего одна задача и не надо управлять промежуточными данными.

```bash

projects/1a/filter_predict.sh
projects/1a/filter.py,projects/1a/predict.py,projects/1a/filter_cond.py,1a.joblib,projects/1a/model.py /datasets/criteo/train-with-id.txt pred_with_filter filter.py predict.py

```

где аргументы:

- файлы, которые надо послать вместе с задачей, через запятую
- путь к входному файлу
- путь к выходному файлу
- имя файла с программой маппера, то есть filter.py.
- имя файла с программой редьюсера, то есть predict.py.

Расчет метрики

Для расчета метрики можете скопировать файл с предсказаниями в домашнюю директорию и запустить скрипт с нужной функцией из пакета sklearn. См. файл local_scorer.py

Проверка

Запустите:

checker.sh 1a

Работа чекера

- проверка наличия всех нужных скриптов. Файл с обученной моделью не должен быть в репо.
- запуск вашего скрипта обучения
- проверка наличия сохраненного файла модели
- запуск ваших скриптов для фильтрации и инференса
- копирование файла с предсказаниями в локальную папку
- расчет метрики

Решение принято, только если вы получили PASSED 1. В противном случае смотрите сообщения чекера.

Подсказки

- Датасет для обучения довольно большой. Чтобы экономить память, рекомендуется не использовать категориальные колонки с большим количеством признаков.
- Ваши контейнеры падают с кодом 137, 143. Это из-за превышения лимитов по памяти.
- Ваши контейнеры падают с кодом 1. Ошибка исполнения в ваших питоновских скриптах. Смотрите логи. Очень помогает команда yarn logs -applicationId application_XXX_XXX после остановки задачи. Далее смотрите на stderr.
- Помните, что чекер запускает собственный скрипт расчета метрики, так что будьте внимательны с форматом результата фильтрации и предсказания.