Стажер - data scientist в группу Яндекс-ЦЕРН

Вам предстоит принять участие в совместных проектах Яндекса и ЦЕРНа - применять современные методы анализа данных к задачам из физики частиц и разрабатывать инструменты, которые позволят делать это другим. Мы занимаемся разработкой алгоритмов машинного обучения, применением машинного обучения для идентификации распадов и поиска неисправностей в детекторе, разработкой инструментов для воспроизводимых исследований. Проектов много и разных: https://github.com/yandexdataschool/students_projects

Если Вы заинтересовались, пожалуйста, отправляйте резюме и решения в форму https://indico.cern.ch/event/525165/registrations/29254/ , по всем вопросам пишите на ysda-cern-interns@yandex-team.ru

Требования:

1. знание математической статистики и теории вероятностей;
2. знание методов анализа данных и машинного обучения;

Как минимум, одно из:

1. опыт использования методов глубокого обучения;
2. опыт использования методов не глубокого обучения;
3. опыт работы с системами анализа больших данных (Hadoop, Spark).

Плюсами будут:

1. знакомство с физикой частиц;
2. опыт работы в Unix/Linux, владение консольными утилитами;
3. знание методов оптимизации;
4. умение программировать на Python;
5. опыт проведения научных исследований;

Условия:

1. длительность стажировки три месяца, точные даты обсуждаются индивидуально;
2. полная занятость, гибкий график;
3. для студентов МФТИ, ВШЭ и физфака МГУ возможно продолжение сотрудничества в формате дипломной работы;
4. и да, мы платим стажёрам зарплату;

Продемонстрируйте свои знания:

Если Вы претендуете на опыт использования методов не глубокого обучения (использовать DL в этой задаче не возбраняется, но ни коим образом не обязательно):

Полтора года назад проводилось kaggle-соревнование про классификацию распадов частиц. Страница соревнования - https://www.kaggle.com/c/higgs-boson

От Вас хочется получить решение задачи с соревнования: от получения данных до отгрузки решения на kaggle. Желательно сделать решение в виде Jupyter тетрадки, хотя

при большом желании вы можете выбрать свой формат. Сильно заморачиваться о качестве решения не нужно, достаточно побить результат 3.4 (baseline).

Также хочется, чтобы Вы ответили на такие вопросы:

1. Почему использование методов машинного обучения оправдано в этой задаче? Как бы Вы ее решали без их использования?
2. Как Вы её решали и почему так? Что пробовали использовать?
3. Насколько хорошо себя ведёт ваша модель и как Вы это оценивали?
4. Как нужно выбирать порог классификации для оптимизации метрики качества AMS? [задача на умение искать в Интернете]

Если Вы претендуете на опыт использования методов глубокого обучения

Вам предлагается решить на выбор одну из 2 задач:

Задача первая: творческий конкурс.

Вам нужно создать сеть, которая сможет порождать новые шрифты. Для ориентира можно использовать данные из датасета NotMnist, однако вы можете использовать любые внешние данные по вкусу. Нужно научить нечто, что сможет как минимум порождать заданную букву в хоть каком-нибудь распознаваемом виде.

Хорошо - если вы сможете порождать шрифты, промежуточные между двумя заданными примерами; похожие на один пример, но не похожие на другой -- и далее всю линейную алгебру шрифтов.

От вас хочется получить описание процесса обучения, его реализацию, обученную модель и демку, которая использует полученную модель для порождения шрифтов. Демка, разумеется, должна быть запускабельной.

Задача вторая: ловля трески.

Есть задача с Kaggle про выявление нежелательного контента среди объявлений на Avito.ru https://www.kaggle.com/c/avito-prohibited-content/

Ваша задача - решить её, минимально используя классические методы обработки текстов и максимально - глубокое обучение (в идеале end-to-end). Как минимум нужно сделать модель, которая побьёт baseline конкурса - 0.92712 - послать её на конкурс и продемонстрировать скриншот + код, который воспроизводит разметку.

Хорошим будет решение, которое даёт precision 0.95. Будет очень круто, если оно будет воспроизводиться целиком менее, чем за 12 часов CPU-time на хорошем пользовательском ноутбуке (трактовка “хорошести” в вашу пользу).

Если Вы претендуете на работы с системами анализа больших данных

Классический WordCount - посчитать частоту слов в большой базе документов. Теперь усложняем - так как есть очень частые слова (например союзы "и" и т.п.) то Reduce для таких ключей будет работать сильно дольше чем для слов со средней частотой. Как решить эту проблему? Напишите код, который бы работал на MapReduce, сравните его производительность с “классическим” WordCount. Насколько результат Вашего сравнения будет применим на кластере в 1000 узлов?