Добрый день, Антон,  
  
Спасибо за ваш интерес к данному направлению и отклик на мое письмо.  
Следующим шагом я предлагаю вам пройти тестовое задание для выявления реального уровня знаний, а также скорости обучения по целевым технологиям:  
  
1. Необходимо сделать простое одностраничное веб-приложение для визуализации данных.  
  
a. взять файл с данными [http://jsonstudio.com/wp-](http://jsonstudio.com/wp-content/uploads/2014/02/world_bank.zip);  
b. загрузить его в MongoDB;  
c. потом поднять Flask приложение и соединиться с MongoDB;  
d. вытащить коллекцию и передать на front-end следующие данные:  
    • project\_name - наименование проекта;  
    • countryname - страна;  
    • lendprojectcost - сумма проекта;  
e. на front-end надо эти данные принять с помощью d3; и  
f. отобразить на карте мира:  
     • choropleth map по количеству инвестированных средств;  
     • в popup отобразить project\_name, countryname и lendprojectcost;  
     • а также сумму по всем проектам в данной стране.  
  
В качестве результата работы обратно вышлите:  
а. скриншот;  
b. код проекта (в zip архиве);  
c. инструкцию по запуску.  
  
Лучше всего это делать из под Ubuntu.  
  
Полезные ссылки:  
Как поставить виртуальную Ubuntu: [http://activewizards.com/blog/](http://activewizards.com/blog/installation-and-running-ubuntu-virtual-box/)  
Документация по Flask: <http://flask.pocoo.org/>  
Первые шаги с MongoDB: [http://activewizards.com/blog/](http://activewizards.com/blog/practical-mongodb-in-10-minutes/)  
Основа для карты на d3.js: <https://datamaps.github.io/>  
  
2. Для проверки вашего уровня технического английского переведите текст на русский или украинский:  
  
Thankfully, the data is already in a simple CSV format and does not require much cleansing or other preparation to be used with Spark MLlib. Later, it will be of interest to explore some transformations of the data, but it can be used as is to start. The covtype.data file should be extracted and copied into HDFS. This chapter will assume that the file is available at /user/ds/. Start spark-shell.  
The Spark MLlib abstraction for a feature vector is known as a LabeledPoint, which consists of a Spark MLlib Vector of features, and a target value, here called the label. The target is a Double value, and Vector is essentially an abstraction on top of many Double values. This suggests that LabeledPoint is only for numeric features. It can be used with categorical features, with appropriate encoding.  
One such encoding is one-hot or 1-of-n encoding, in which one categorical feature that takes on N distinct values becomes N numeric features, each taking on the value 0 or 1. Exactly one of the N values has value 1, and the others are 0. For example, a categorical feature for weather that can be cloudy, rainy, or clear would become three numeric features, where cloudy is represented by 1,0,0; rainy by 0,1,0; and so on. These three numeric features might be thought of as is\_cloudy, is\_rainy, and is\_clear features. Another possible encoding simply assigns a distinct numeric value to each possible  
value of the categorical feature. For example, cloudy may become 1.0, rainy 2.0, and so on.

Хорошо что данные уже в простом CSV формате и нет обязательной необходимости в других преабразованиях данных для использования их в Spark Mllib. Позже, будет интересно исследовать некоторые трансформации данных, но мы можем использовать такие данные для начала. Запустите spark-shell.

Spark Mllib абстракция для свойства вектора известного как LablePoint(метка уровня), которая содержит Spark Mllib Vector внутри которого свойства и конечные значения, вызываемые для уровня. Целевое значение имеет тип double, а вектор это абстракция над всеми значениями. Такое решение как LablePoint подходит только для цифровых свойств(функций). Оно может быть использовано с группой свойств, в определенной кодировке. Одной из таких кодировок является еденичная кодировка или от 1 до n, это значит кодировка принимающая свойства с N значениями, каждое из которых должно быть от 0 до 1. Вообще одно из N значений должно быть 1 а все остальные 0. Для примера, группа свойств погоды может иметь такие значения облачно, дождливо или ястно значения могут быть преабразованы в числовые где облачно будет представленно как 1,0,0; дождливо как 0,1,0; и так далее. Эти три цифровых свойства могут быть преабразованны в свойства облачно\_ли , дождливо\_ли, небочистое\_ли. Другие возможные кодировки легко принимать как отдельно возможные числовые значения категорий свойств. Например, облачно 1.0, дождливо 2.0 и так далее.

For start Tornado application python version must be >= 3.5

Install MongoDB:

sudo apt-key adv --keyserver hkp://keyserver.ubuntu.com:80 --recv 0C49F3730359A14518585931BC711F9BA15703C6  
Create a sudo nano /etc/apt/sources.list.d/mongodb-org-3.4.list

echo "deb http://repo.mongodb.org/apt/debian jessie/mongodb-org/3.4 main" | sudo tee /etc/apt/sources.list.d/mongodb-org-3.4.list

sudo apt-get update

sudo apt-get install -y mongodb-org

pip install -r requirements.txt

sudo service mongod start (sudo service mongod stop, sudo service mongod restart)

mongoimport --db testdb --collection world --file world\_bank.json

Start Flask dvelopment server

export FLASK\_APP=hello.py

python -m flask run