Data Science Camp 2016

Session Notes

## Managing and Versioning Machine Learning Models in Python

Simon Frid, github.com/fridiculous

Overview

1. Motivation
   1. Image Recognition Use Case
   2. Ad Conversion Use Case
   3. Fraud Prediction Use Case
2. Strategies and Design Considerations
   1. Data Science Workflow
   2. What Can We Learn from Software Version Control
3. Python Tools
4. Demos
   1. Estimators
   2. Trusted Analytics

* Use Case 1: Car Production
  + Iterative process
* Use Case 2: Selling Student Loans
  + Training
* Use Case 3: Payment Gateway

“There are practical little things in housekeeping which no man really understands.” – Eleanor Roosevelt

* [Data Science Workflow Image]
  + Blue = Business
  + Yellow = Sandbox
  + Red = Production

|  |  |
| --- | --- |
| Concept in SVC | Tech Needed |
| Repository | Persistence & Serialization |
| Versioning | Indexing & Hashing |
| Commits, Tags and Labels | Attributes & Tags |
| Push, Pull and Checkout | API to Persist and Retrieve |
| Diff | ☺ |

### Python Tools

Algorithm Options

Persistence Layer Options

* S3
* GitLFS
* Elasticsearch and Document-based Stores
* Docker
* Pachyderm

Serialization Options

* cpickle (py2) and pickle (py3)
* Sklearn.joblib
* Dill, cloudpickle and picklable-itertools
* PMML via jpmml-sklearn
* What about transformer pipelines?

Indexing & Hashing

* Hashing the model
* Hashing the data
* Relational DB Table for Look Up
* Key Value Stores like Redis, Dynamo

Labels

* Semantic Versioning, Major.Minor.Patch
* Tags (Django-taggit)
* Store MetaData, create\_dates, relationships between models
* Notes and learning from (Humans in the Loop)

API… components

* Customer using and ORM/DAL like Django and sqlalchemy
* SaaS & PaaS – Turi, Science Ops, Prediction IO, Azure ML
* Asynch Tasks – Airflow, Luigi, Celery
* Flows using Docker and Pachyderm

Estimators

Estimators Demo

Running Models and their Apps in Production with Trusted Analytics Platform (TAP)

* TAP
* Jupyter Notebook can be accessed through it

### Scalable Computing in R on Spark

Motivation

* Training hierarchical time series forecasting models require searching through a large parameter space
* Using HDInsight cluster

Middle-Out Approach

Search space complexity

Iterative optimization

* Baseline setting: MAPE = 9.1%
* Able to reduce MAPE to ~6.8%

Results

* 1st graph error update: x = MAPE, y = count
  + Can get really large if wrong param or hierarchy
* MAPE improved from 9.1% -> 6.8%
* Execution improved from 41 days -> ~1 day
* Don’t need to modify R, just change a compute context

Conclusion

Questions

* Using EvoSparkR
* Can get from MSDN Subscription and Dev Essentials

### Traffic Sign Recognition with Tensorflow

Waleed Abdulla

Jupyter Notebook

* <https://github.com/waleedka/traffic-signs-tensorflow>

Technologies

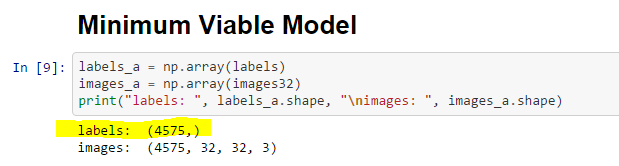
* Jupyter
* Tensorflow
* Python

Training Data Set

* Fixes
  + Add more padding
  + Crop them down to remove padding
* Both are a lot of work
* His research resizes to the same size



* “If you understand the data that helps w/ understanding the results.”
* Images are resized to 32x32 pixels
* RGB Range: 0 (black) to 1 (white)
* Tensorflow takes numpy arrays at input
* Labels are what we want the model to learn from



Fully Connected Network

* Input Layer -> Hidden Layer 1 -> … -> Hidden Layer n -> Output Layer
* Training code: Optimized C++ and CUDA (for GPU)

What’s next?

* Convolutional NN
* Scaling the dataset
* Data augmentation
* Image detection

### Industrial IoT, Discuss Current State of IIoT Analytics, AKA Digital Twin

Robert Benson­­­

* Cloud Foundry, leading open source cloud