OpenBLAS & numpy setup

OpenBLAS & Numpy test (https://hunseblog.wordpress.com/2014/09/15/installing-numpy-and-openblas/)

Task	Not linked	OpenBLAS	OpenBLAS + Lapack
dotted two (1000,1000) matrices	2,838 ms	53.9 ms	17.5 ms
dotted two (4000) vectors	6.95 us	3.69 us	4.89 us
SVD of (2000,1000) matrix	10.883 s	10.739 s	0.744 s
Eigendecomp of (1500,1500) matrix	37.648 s	34.339 s	7.932 s

Check install:

```
import numpy as np
np.__config__.show()
```

python3.4 fixes some of the problems (https://github.com/numpy/numpy/blob/v1.12.1/site.cfg.example#L119) with multithreaded OpenBLAS

Experiments 2018-02-04

Summary: A lot of machine learning tools depend on matrix manipulation libraries, e.g. numpy. In a basic configuration it uses CPU for linear algebra computations, such as matrix multiplication, SVD or Eigenvalues decomposition. OpenBLAS speeds computations 4-10x via Fortran binding.

Python + NumPy

Dell XPS, VirtualBox VM Ubuntu Xenial, 10G RAM, 8CPU.
Numpy: 1.14.0, OpenBLAS: 0.2.20

Task	Python Default	Python with OpenBLAS
dotted two (1000,1000) matrices	42.3 ms	55.8 ms
dotted two (4000) vectors	4.96 us	4.84 us
SVD of (2000,1000) matrix	1.164 s	1.069 s
Eigendecomp of (1500,1500) matrix	12.110 s	**3.455 s**

R

Task	R Vanilla	R OpenBLAS
dotted two (1000,1000) matrices	87.1 ms	56.6 ms
dotted two (4000) vectors	27.9 us	30.06 us
SVD of (2000,1000) matrix	9.489 s	2.08 s
Eigendecomp of (1500,1500) matrix	26.175 s	10.06 s

R vanilla is significantly slower than numpy.

Hackathon results:

1. Benchmarked NumPy and R with and without OpenBLAS
2. Released example Ansible repository: https://github.com/slothai/ansible-example (https://github.com/slothai/ansible-example)