DevOps Workshop

PRESENTED BY -
Aayush Agrawal

Aayush Agrawal

Education -

- MS in Business Analytics, Carlson School of Management, University of Minnesota, 2017
- B.Tech in Electrical Engineering, Malaviya National Institute of Technology, India, 2013

Experience –

- >5 years in Data science, Currently working as a Data scientist at Land O' Lakes, Inc.
- Moderator and rank 3rd at https://www.analyticsvidhya.com/
- Kaggle Expert - https://www.kaggle.com/aayushmnit

Reach me out at (@aayushmnit) –

Email - aayushmnit@gmail.com

LinkedIn - https://www.linkedin.com/in/aayushmnit/

Github - https://aayushmnit.github.io/

Medium - https://medium.com/@aayushmnit

Agenda

- **Model Building using cookbooks**
- **Version control by GIT**
- **Deploying models as API using Flask**
- **Packaging applications using Docker**
- **Deploying application in cloud**

DevOps at a glance

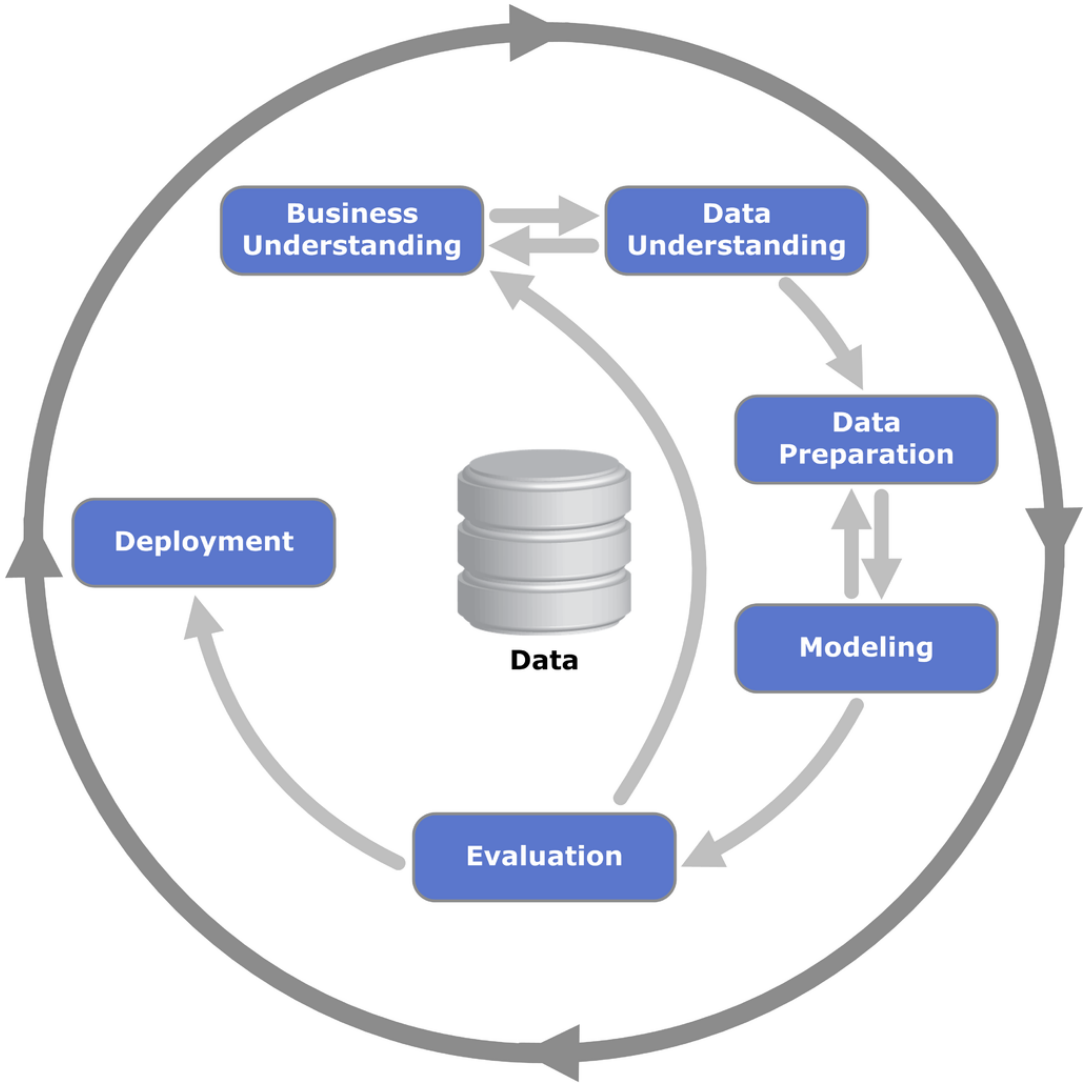
DevOps-

- Continuous improvement(Test and learn)
- Improve the software delivery process (stakeholder management)
  - Reduce rework
  - Reduce overhead by automation

DevOps in data science -

- Neat, consistent, reliable & structured code
- Deploy model with API
- Use containers
- Use orchestration tools
- Monitor all level of stacks

CRISP DM – Best methodology for Data science



Business Understanding
Translation of project objectives
and requirements from a business perspective to data Science problem

Data Understanding
Data collection, quality checks, exploration, hypothesis testing

Data preparation
Process of converting raw data to model ready data

Modeling
Building advance analytics/ machine learning model

Evaluation
Model quality assessment and results interpretation

Deployment
Automation of data and modeling pipelines

Use of Cookbooks in Data science

- Many of the data science task are quite repeatable and highly generalizable
- 60% of the time writing code is wasted on functions/program which a data scientist have already written

Reusability
- Less syntax to remember
- DRY (Don't repeat yourself)

Modularity
- Cleaner and well written code
- Reliable performance by multiple re-use in different scenario

Faster execution
- Less code to write (10Xer)
- Consistent output to analyze

Customizable
- Appending/Removing functionality
- Customizable documentation

My cookbooks overview

Link to git repo - https://github.com/aayushmnit/cookbook

Generic Preprocessing - Helper functions to do EDA, missing value analysis & treatment and generic preprocessing like (scaling, encoding etc.)

Machine learning - Classification - Helper functions to solve classification type problems in machine learning. It contains codes for holdout/cross validation, model explanation codes (LIME and variable importance plot), and codes for general classification algorithms (Xgboost, LightGBM, Extra trees, random forest, logistic regression, decision trees, K-nearest neighbors and SVM)

Machine learning - Regression - Helper functions to solve regression type problems in machine learning. It contains codes for holdout/cross validation, model explanation codes (LIME and variable importance plot), and codes for general regression algorithms (Xgboost, LightGBM, Extra trees, random forest, linear regression, regression trees, K-nearest neighbors and SVM)

Recommender systems - Helper functions to build recommender systems using Matrix factorization using LightFM package.

Natural language processing - Helper functions for NLP text processing and analysis like N-grams, word cloud, tokenization, lowercasing, punctuation/stopwords removal, stemmer/lemmatizer, TF-IDF & count vectorizer

Agenda

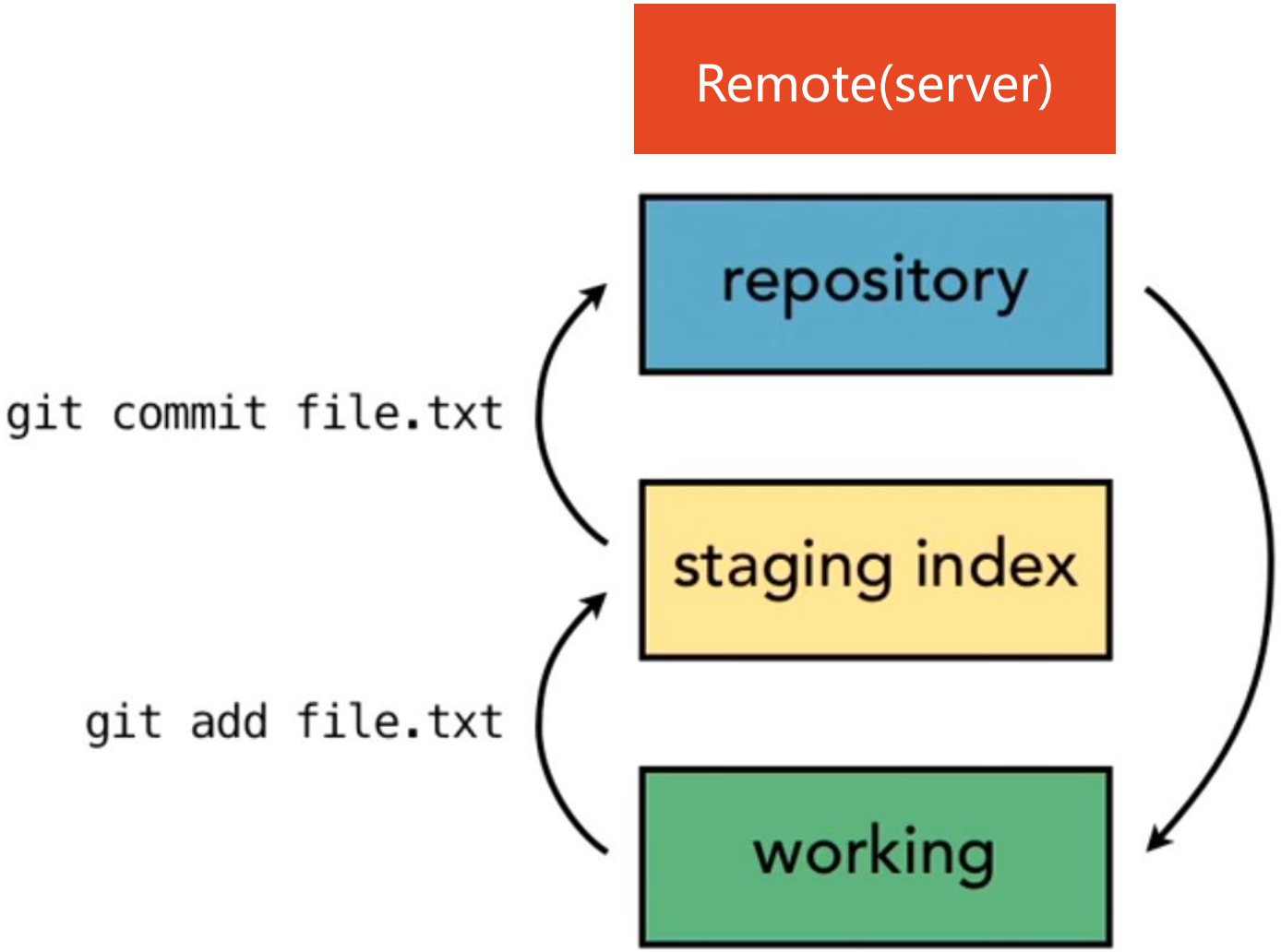
- Model Building using cookbooks
- **Version control by GIT**
- Deploying models as API using Flask
- Packaging applications using Docker
- Deploying application in cloud

Who should use Git?

- Anyone wanting to track edits
  - Review a history log of changes made
  - View differences between versions
  - Retrieve old versions
- Anyone needing to share changes with collaborators
- Anyone not afraid of command line tools

https://www.linkedin.com/learning/git-essential-training/who-should-use-git

Git follows a three tree architecture



https://www.linkedin.com/learning/git-essential-training/exploring-the-three-trees-architecture

Useful Links

- **Git essentials training - https://www.linkedin.com/learning/git-essential-training**
- **Git cheat sheet**

Agenda

- Model Building using cookbooks
- Version control by GIT
- **Deploying models as API using Flask**
- Packaging applications using Docker
- Deploying application in cloud

Flask is a micro service web platform framework

Many machine learning models are used real time at the heart of any data product.

Options to implement Machine Learning models –

Option 1: Rewriting the whole code in the same language as product
- Very hard and time consuming process

Option 2: API-first approach
- Web APIs have made it easy for cross-language applications to work well
- Software product can just call the API with bunch of parameters to get result regardless of backend language

Agenda

- Model Building using cookbooks
- Version control by GIT
- Deploying models as API using Flask
- **Packaging applications using Docker**
- Deploying application in cloud

Docker is a containerization platform

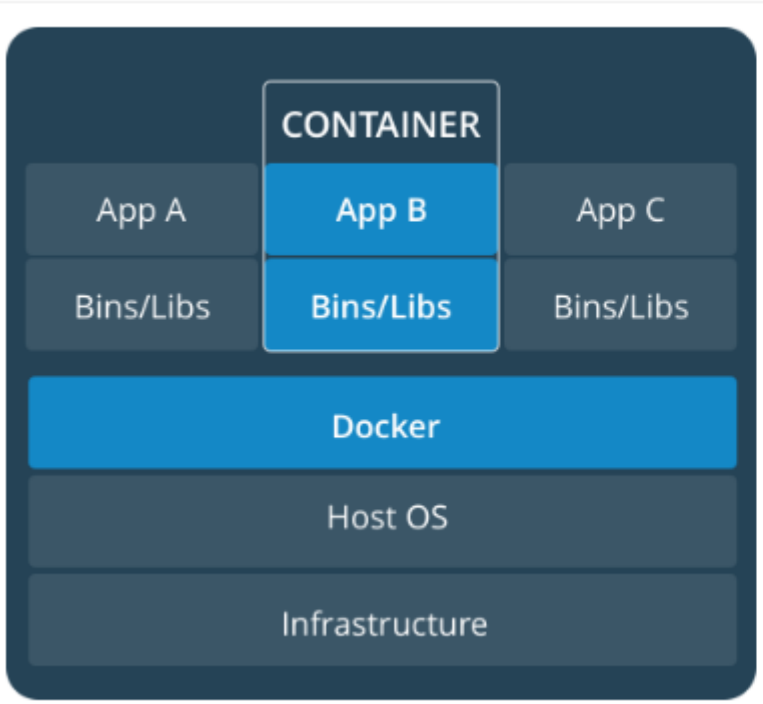
Docker is a platform for developers and sysadmins to develop, deploy, and run applications with containers.

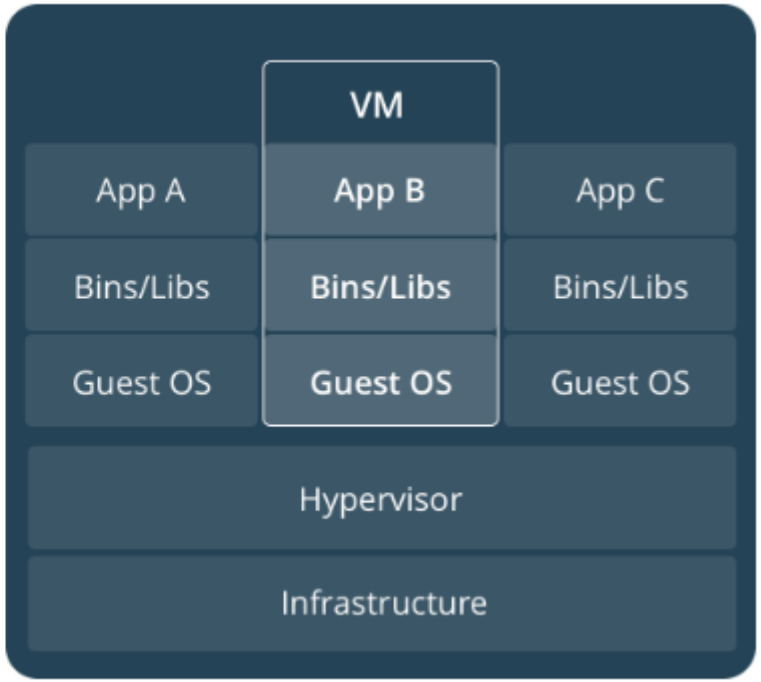
Containerization is increasingly popular because containers are:

- **Flexible:** Even the most complex applications can be containerized.
- **Lightweight:** Containers leverage and share the host kernel.
- **Interchangeable:** You can deploy updates and upgrades on-the-fly.
- **Portable:** You can build locally, deploy to the cloud, and run anywhere.
- **Scalable:** You can increase and automatically distribute container replicas.
- **Stackable:** You can stack services vertically and on-the-fly.

Image - An image is an executable package that includes everything needed to run an application--the code, a runtime, libraries, environment variables, and configuration files
Container - A container is a runtime instance of an image--what the image becomes in memory when executed

Container is light weight because it doesn't require an OS





Creating your own docker file

https://docs.docker.com/get-started/part2/#dockerfile

```
# Use an official Python runtime as a parent image
FROM python:2.7-slim

# Set the working directory to /app
WORKDIR /app

# Copy the current directory contents into the container at /app
ADD . /app

# Install any needed packages specified in requirements.txt
RUN pip install --trusted-host pypi.python.org -r requirements.txt

# Make port 80 available to the world outside this container
EXPOSE 80

# Define environment variable
ENV NAME World

# Run app.py when the container launches
CMD ["python", "app.py"]
```

Basic Docker commands

Basic commands -

- List docker images: "*docker images*"
- To run a docker image: "docker run --<option> <image_name>
- Delete a docker image: "*docker rmi <image ID>*"

- List all the containers :"*docker container ls -la*"
- Stop container from running: "*docker container stop <container ID>*"
- Stop container from running: "*docker rm <container ID>*"

To build a docker image –

- cd to the folder DOCKERFILE
- *Run the command "docker build -t <image_name>"*

To push an image to DockerHub -

- Create a repo on docker hub
- Login using DockerHub credentials: "*docker login --username=<user_id>*"
- Run command "*docker tag <image_id> <user_name>/<repo_name>:<tag>*"
- Run command "*docker push <user_name>/<repo_name>*"

Scaling with Kubernetes

- Kubernetes is an open-source system for automating deployment, scaling, and management of containerized applications.

Kubernetes can perform simple task –
- Starting and stopping containers
- Determine where to run containers
- Checking health of containers
- Restarting or replacing unhealthy containers

Master node runs -
- Kube-apiserver
- Etcd(Key-value) for cluster data
- Kube-scheduler for deciding where to run a container
- Kube-controller-manager controls replication processes and endpoints
- Cloud controller manager which interacts with cloud services

Minion node runs –
- Kublet – to make sure containers run inside a pod
- Kube-proxy – to manage network operations
- Container-runtime – manages execution of containers

kubernetes

Some good resources on topic discussed

Git essential training- Lynda

Docker –

- Docker official tutorial – Link
- How Docker Can Help You Become A More Effective Data Scientist – Hamel Husain
- A Step Towards Reproducible Data Science : Docker for Data Science Workflows – Analytics Vidhya

API using FLASK –

- Tutorial to deploy Machine Learning models in Production as APIs (using Flask) – Analytics Vidhya
- How to Deploy Keras Models to Production – Siraj Raval
- Flask Tutorial – Link

Deployment -

Deployment using Flask, docker and Kubernetes – Medium
Deploy Your First Deep Learning Model On Kubernetes With Python, Keras, Flask, and Docker - Medium
Deployable machine learning for data science – Lynda

Thank You!