How to Quickly Deploy Machine Learning Model Anywhere

Dr. Erekle Magradze

Ilia State University
Computer Science Program

Challenge

- We moved from the age of discovery to the age of implementation.
- Many great startups launched in the garages – obviously having a garage is not enough :)
- It’s not only enough to have a nice Jupyter notebook to demonstrate the results – the expectation is to have the MVP (Minimal Viable Product) with the ML model, as soon as possible – that’s your competitive advantage;

How to do that???

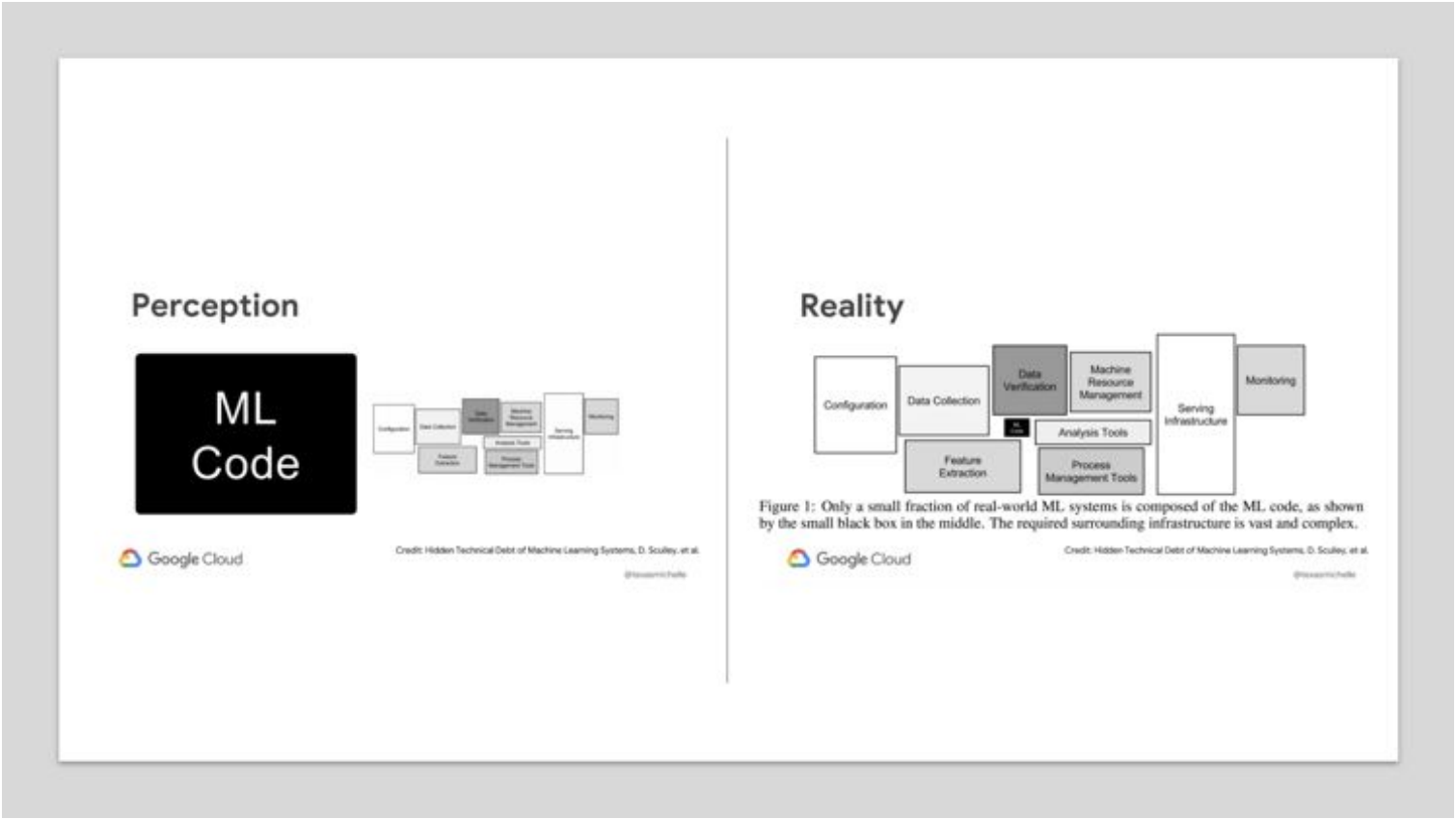
Problems

You have trained the model – now you'd like to show it to your customer over the ocean – asking to do the following steps

- Install Python/go/Julia/Rust/R
- Install all the necessary packages and libraries
- Setup environment variables (don't forget to open this and that ports)
- Copy my code to your machine – don't forget to split them in several files
- Run the code with required parameters

Eventually

Perception vs Reality



From Perception to Reality (Slides extracted from [Strata Data Conference — Kubeflow explained: Portable machine learning on Kubernetes])

I am a Software or ML engineer ...

And I don’t care about the technical stuff required for the deployment - well, nowadays it’s not like that anymore ;)

- Use containerization technology – in particular the docker https://www.docker.com/ use community edition, it’s free
- Once installed and it’s running you can run the following command in your terminal (in case of Mac and Linux) or in your powershell (in case of windows)

  Here is the command

  docker run --rm -p 5000:5000 datascienceexplorer/classifier

  After running this command go to the following url

  http://localhost:5000/apidocs/

- This is a ready application with Machine Learning model in it – based on IRIS dataset, you can try it over the swager i.e. the link shown above

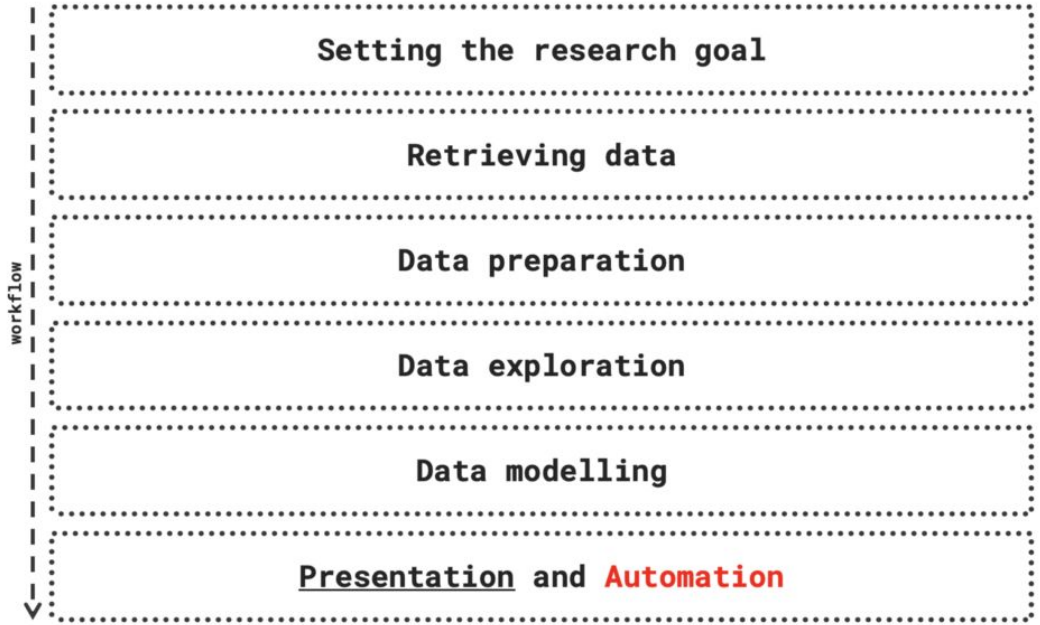
OKay, but what exactly is docker?

Docker is your friend :)

- Docker is a containerization tool, working and running on majority of operating systems;
- It helps to pack all the necessary packages and dependencies in one file – called container image and deploy it on any OS without installation of additional software or opening the ports and doing sysadmin or system integrator tasks – it’s all there for free, out of the box
- How it’s possible because of containerization concept

Containers? Long story - short

Steps of ML project



It’s great when you have lots of time to wait for ANN or CNN model training - automate it and run it, that’s fine!

When you need to give a quick response - we need to keep the model in memory

In case of python use pickle - object serialization package

```
1  import pickle
2
3  with open('./model.pkl', 'wb') as model_pkl:
4    pickle.dump(knn, model_pkl)
```

Let’s have a look to the Jupyter notebook

https://github.com/hermag/docker-session/blob/master/SupervisedMachineLearningClassification.ipynb

- More or less all should be clear if you are doing ML or Data Science
- The most important part for us is this

Save Model

In [52]:
```
import pickle
```

In [55]:
```
with open('./model.pkl', 'wb') as model_pkl:
    pickle.dump(knn, model_pkl)
```

How to use pickled model

We can write simple code to load the pickled model and use it

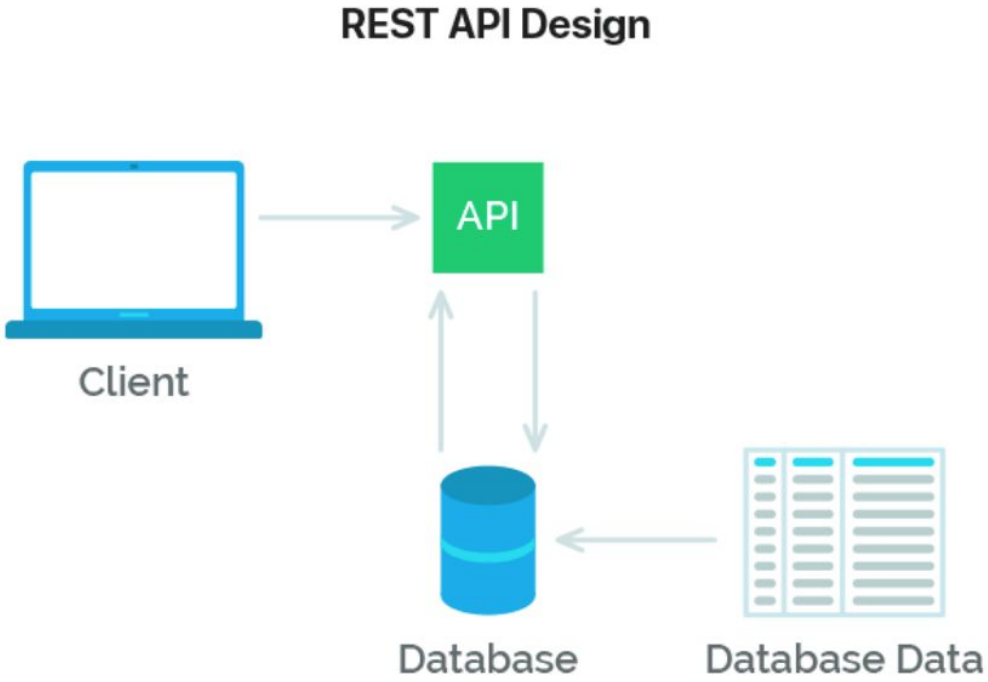
```
import pickle
# Import all the packages you need for your model below
import numpy as np
from sklearn.neighbors import KNeighborsClassifier

# Load the model into memory
with open('./model.pkl', 'rb') as model_pkl:
    knn = pickle.load(model_pkl)

# Unseen data (create a new observation for testing)
unseen = np.array([[3.2, 1.1, 1.5, 2.1]])
result = knn.predict(unseen)

# Print result to the console
print('Predicted result for observation ' + str(unseen) + ' is: ' + str(result))
```

But you might face the following problem

To solve this we need to install the missing package

```
Traceback (most recent call last):
  File "main.py", line 4, in <module>
    from sklearn.neighbors import KNeighborsClassifier
ImportError: No module named sklearn.neighbors
```

This is not enough - we need a service



In case of python we can use Flask

```python
import pickle
# Import all the packages you need for your model below
import numpy as np
import sys
from sklearn.neighbors import KNeighborsClassifier
# Import Flask for creating API
from flask import Flask, request

# Load the trained model from current directory
with open('./model.pkl', 'rb') as model_pkl:
    knn = pickle.load(model_pkl)

# Initialise a Flask app
app = Flask(__name__)

# Create an API endpoint
@app.route('/predict')
def predict_iris():
    # Read all necessary request parameters
    sl = request.args.get('sl')
    sw = request.args.get('sw')
    pl = request.args.get('pl')
    pw = request.args.get('pw')

    # Use the predict method of the model to
    # get the prediction for unseen data
    unseen = np.array([[sl, sw, pl, pw]])
    result = knn.predict(unseen)

    # return the result back
    return 'Predicted result for observation ' + str(unseen) + ' is: ' + str(result)


if __name__ == '__main__':
    app.run()
```

Finally the dockerfile

To wrap the flask application we finally need the dockerfile to build the docker image

```
1  FROM python:3.7
2  WORKDIR /app
3  EXPOSE 5000
4
5  COPY ./requirements.txt .
6  RUN pip install -r requirements.txt
7
8  COPY . .
9  CMD python main.py
```

How our folder structure should look?

```
.
├── Dockerfile
├── README.md
├── SupervisedMachineLearningClassification.ipynb
├── main.py
├── model.pkl
└── requirements.txt
```

To build the container image: ***docker build -t mymlmodel***

To start the container using the built image: ***docker run -name contname -d mymlmodel***

To expose the container ports: ***docker run -name contname -p 5000:5000 -d mymlmodel***

To attach the storage: ***docker run -name contname -v localdrive:/container/path -p 5000:5000 -d mymlmodel***

All materials are available here

https://github.com/hermag/docker-session