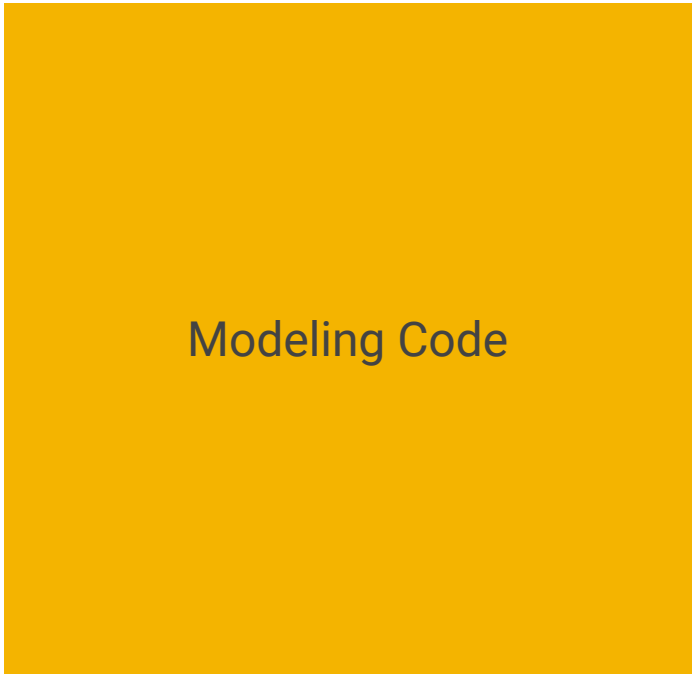
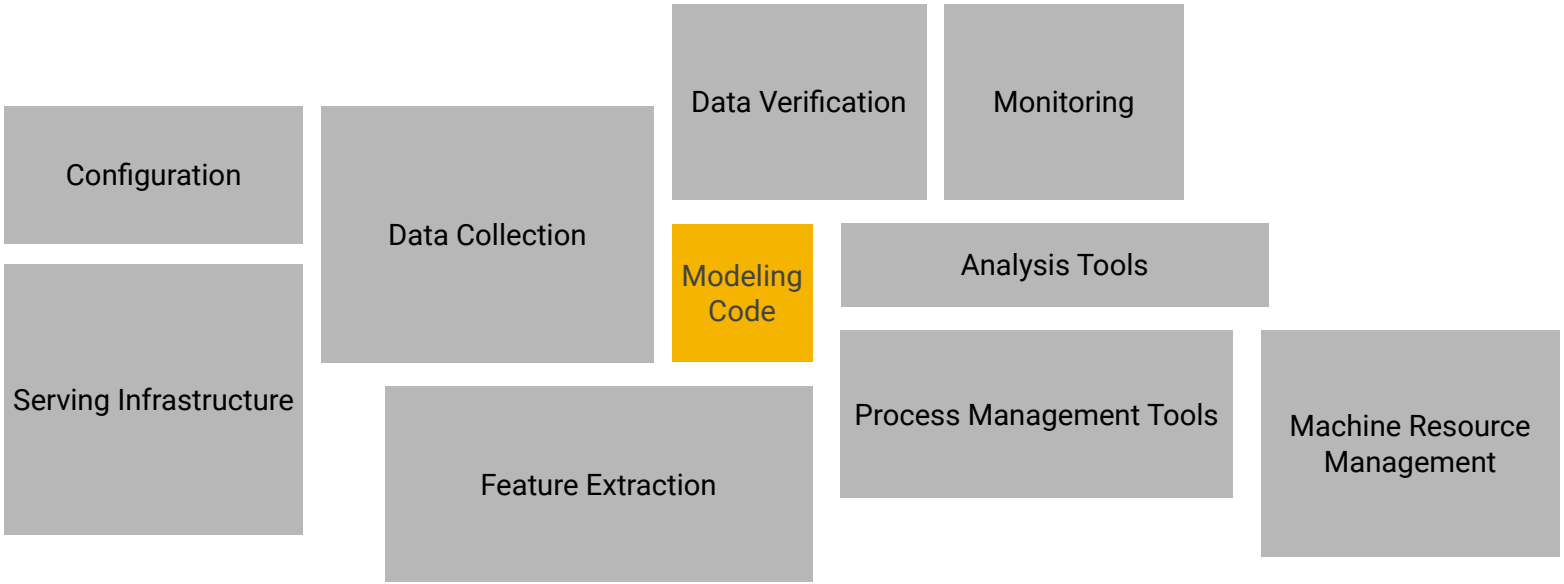
Copyright Notice

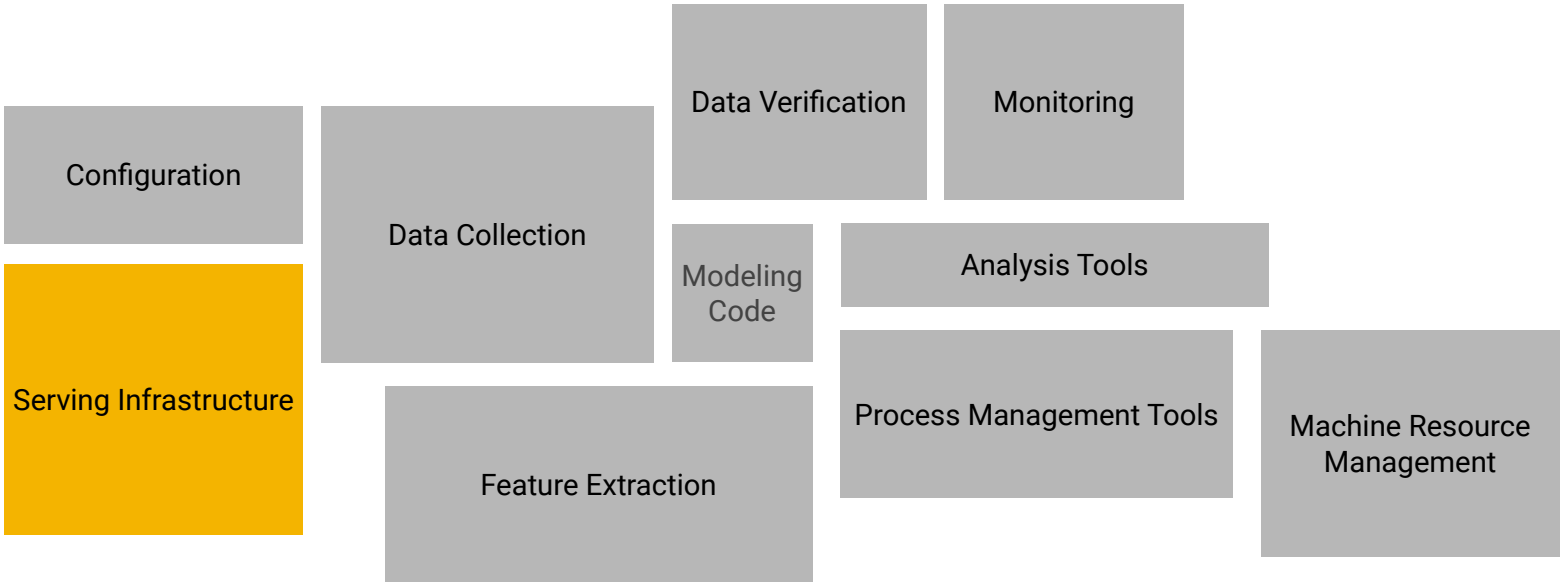
Building Models is just a small part of ML...

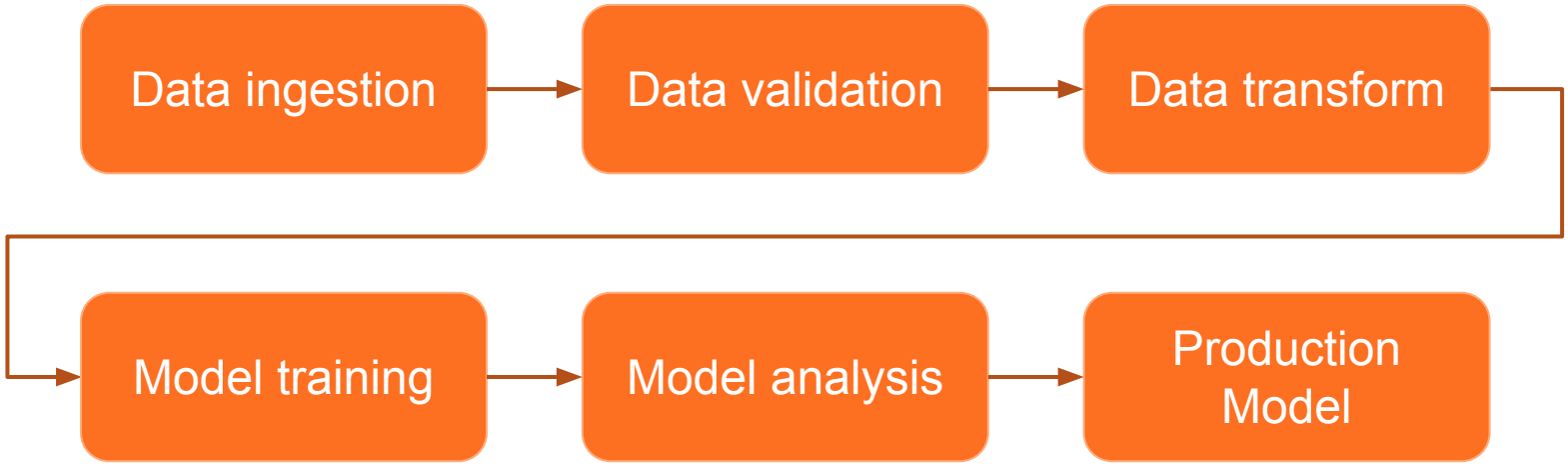


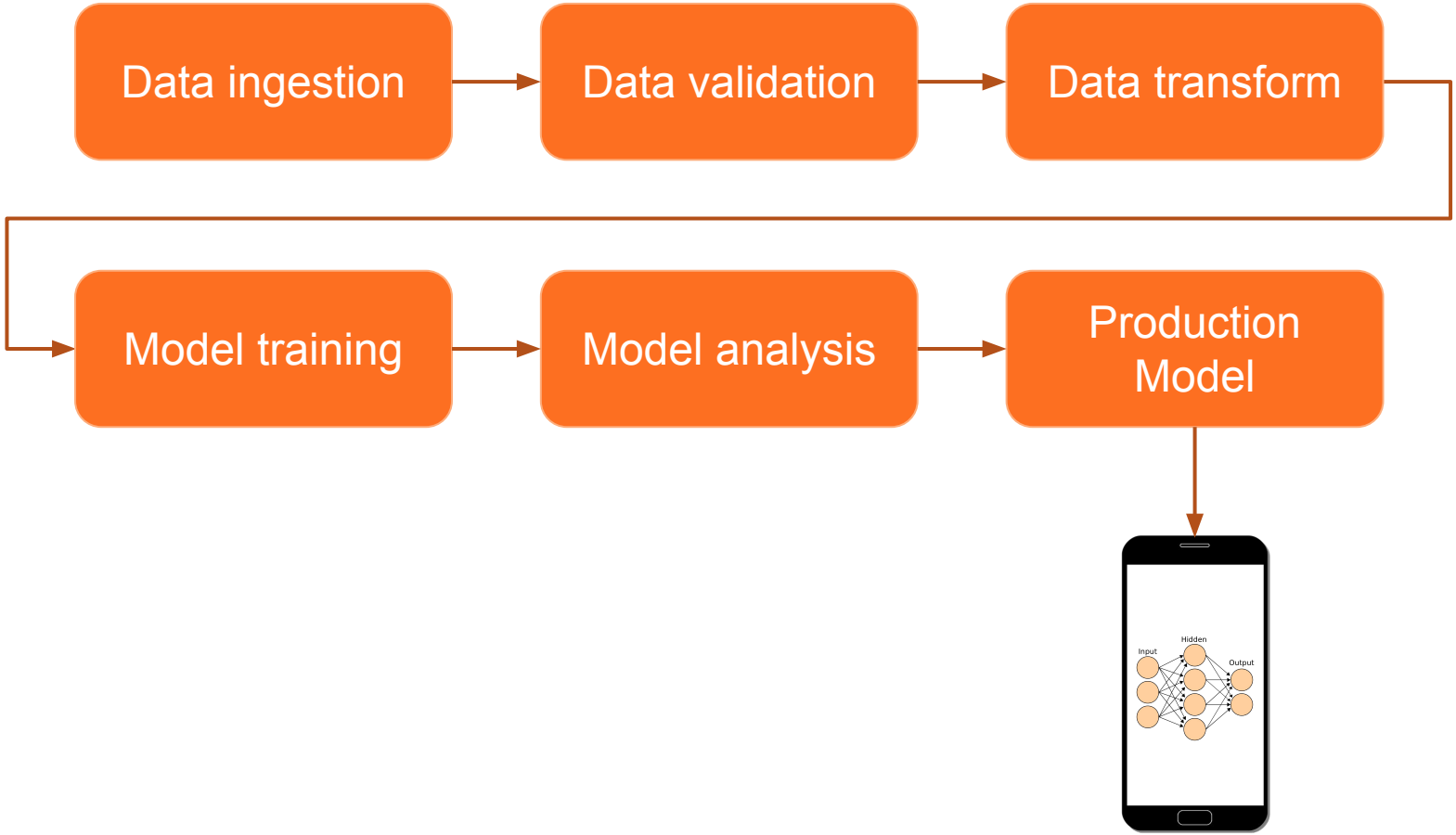
… a production solution requires so much more

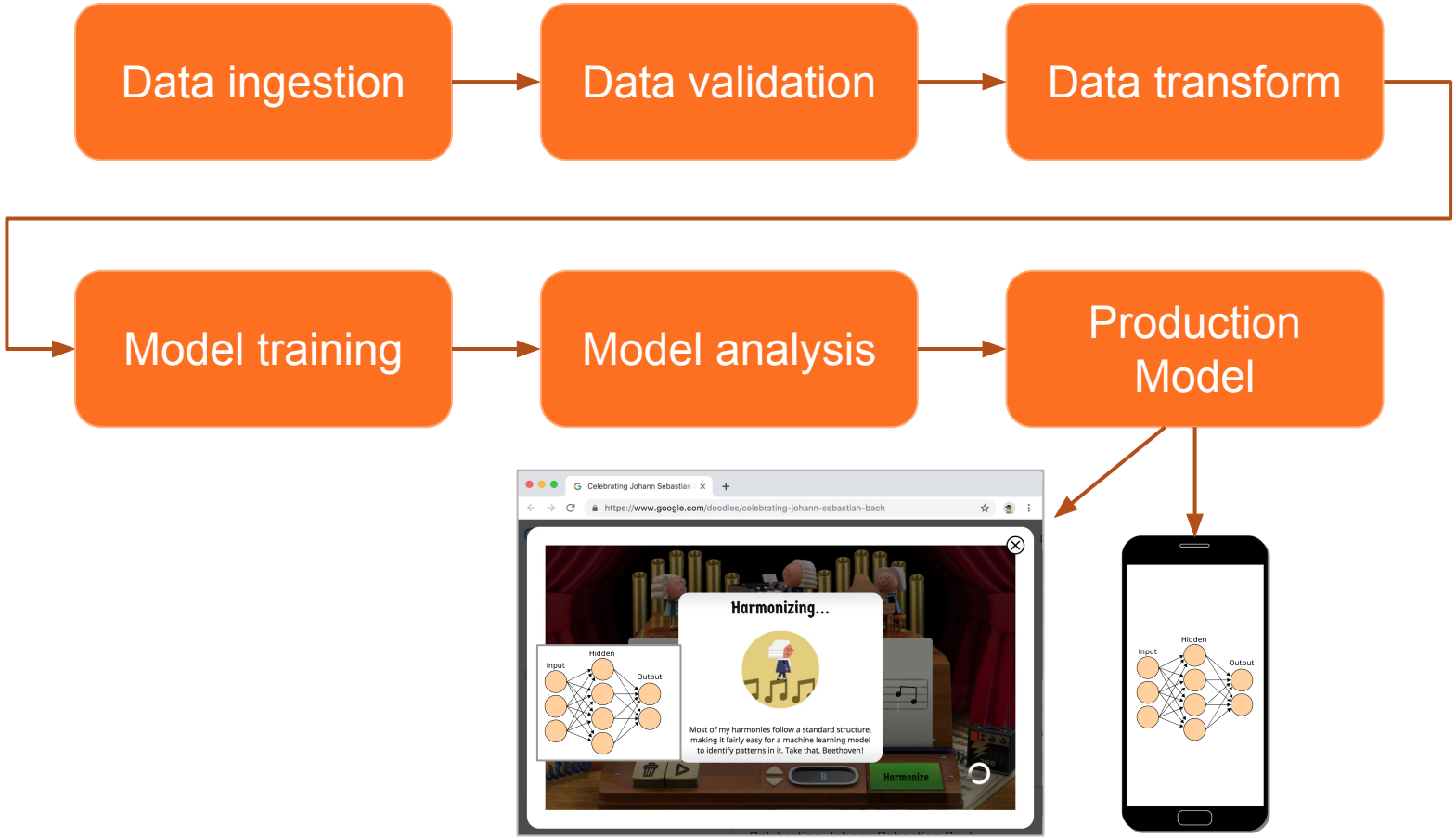


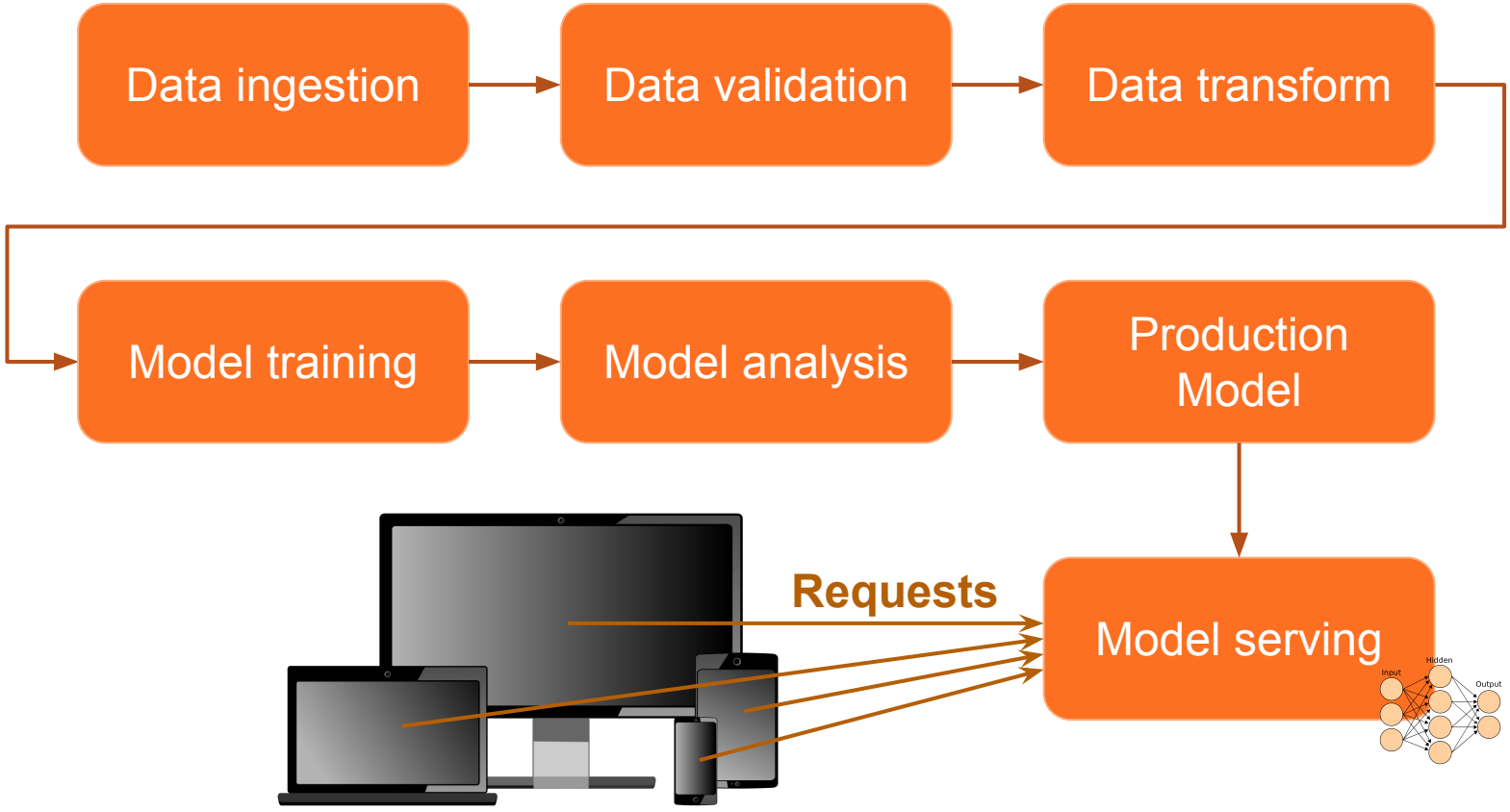
So this week we will focus on Serving...

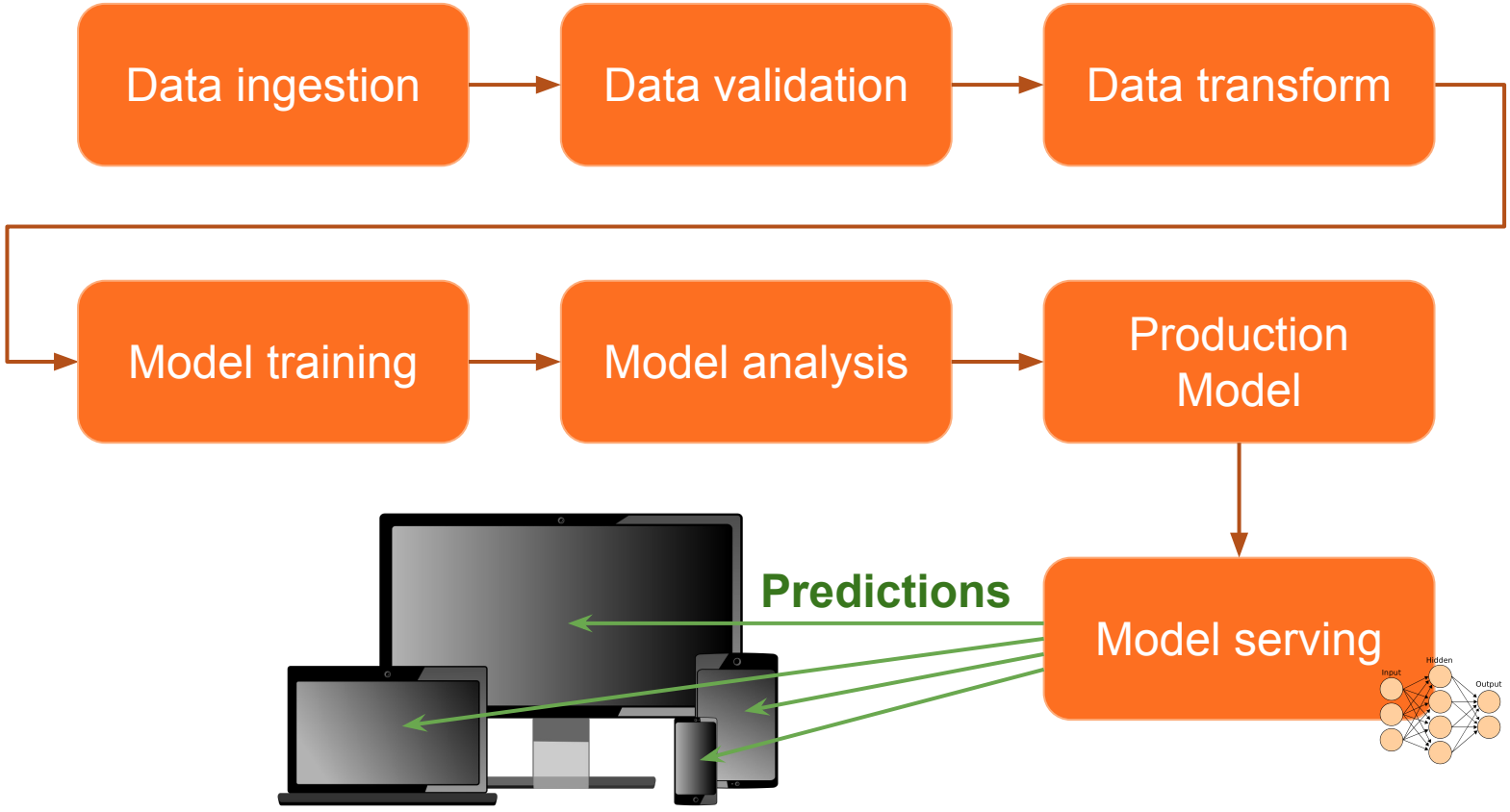


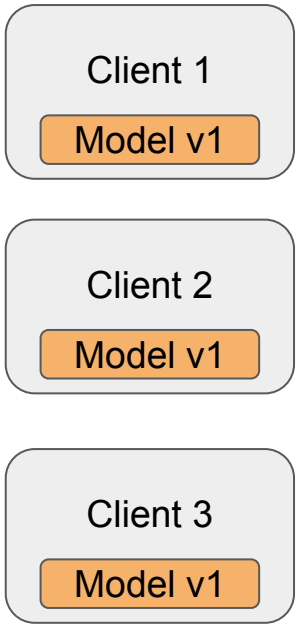

Data ingestion
Data validation
Data transform
Model training
Model analysis
Production Model

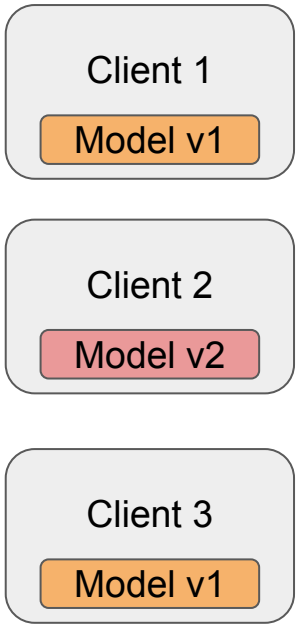

Data ingestion
Data validation
Data transform
Model training
Model analysis
Production Model
Input
Hidden
Output

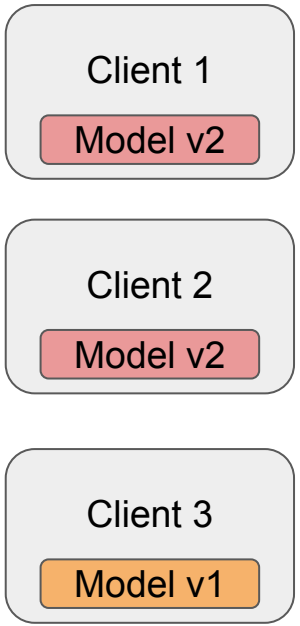

Data ingestion
Data validation
Data transform
Model training
Model analysis
Production Model
Celebrating Johann Sebastian
https://www.google.com/doodles/celebrating-johann-sebastian-bach
Harmonizing...
Input
Hidden
Output
Most of my harmonies follow a standard structure, making it fairly easy for a machine learning model to identify patterns in it. Take that, Beethoven!
Harmonize
Input
Hidden
Output

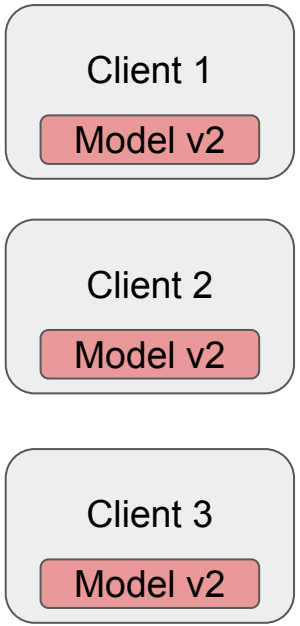

Data ingestion
Data validation
Data transform
Model training
Model analysis
Production Model
Requests
Model serving
Input
Hidden
Output

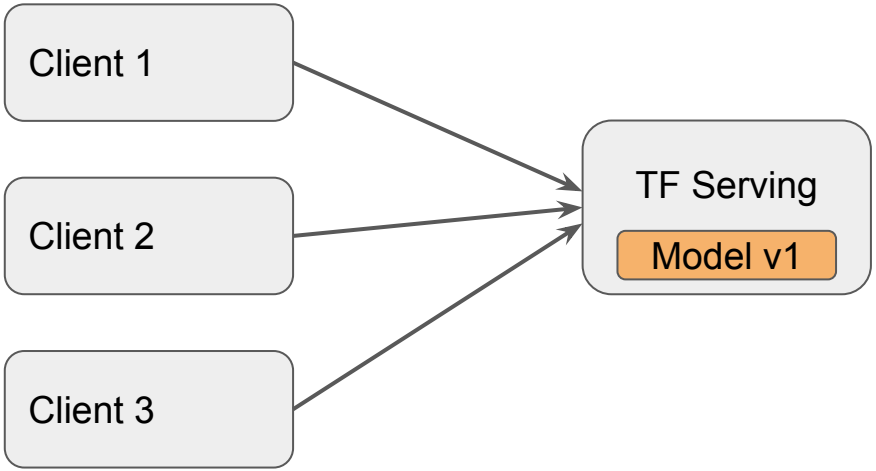

Data ingestion
Data validation
Data transform
Model training
Model analysis
Production Model
Predictions
Model serving
Input
Hidden
Output

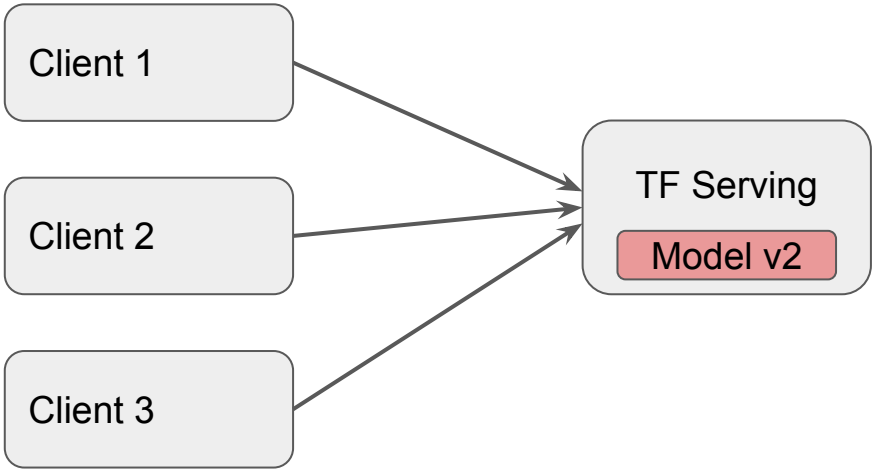

Client 1
Model v1
Client 2
Model v1
Client 3
Model v1

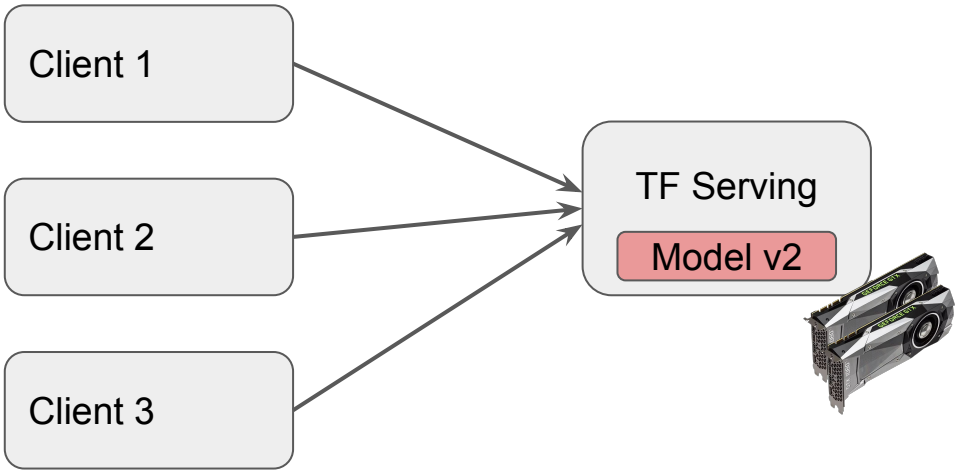

Client 1
Model v1
Client 2
Model v2
Client 3
Model v1

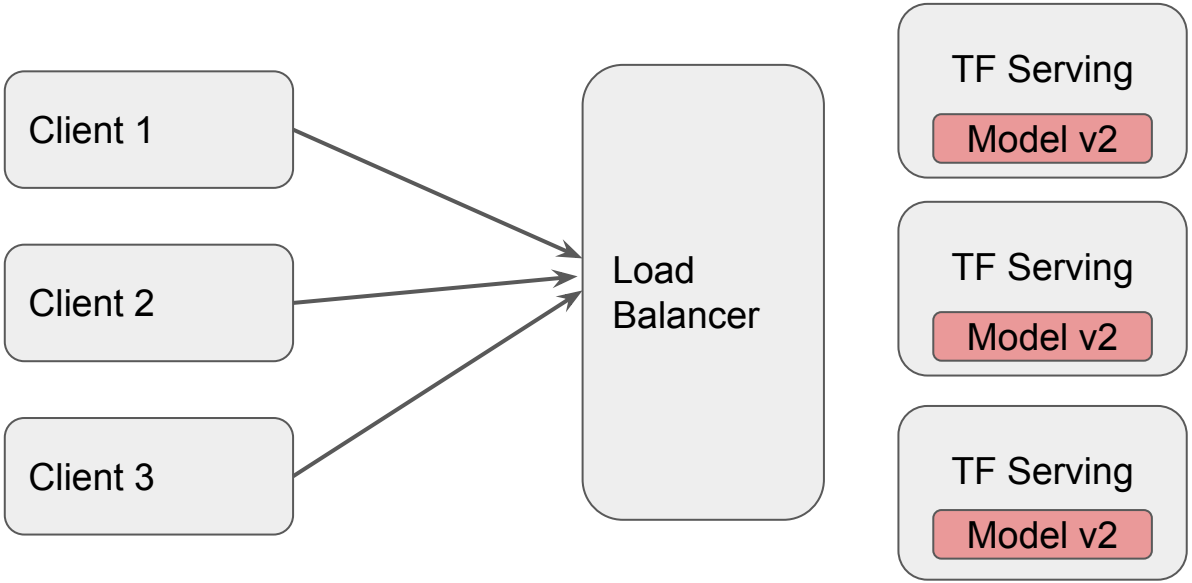

Client 1
Model v2
Client 2
Model v2
Client 3
Model v1


Client 1
Model v2
Client 2
Model v2
Client 3
Model v2

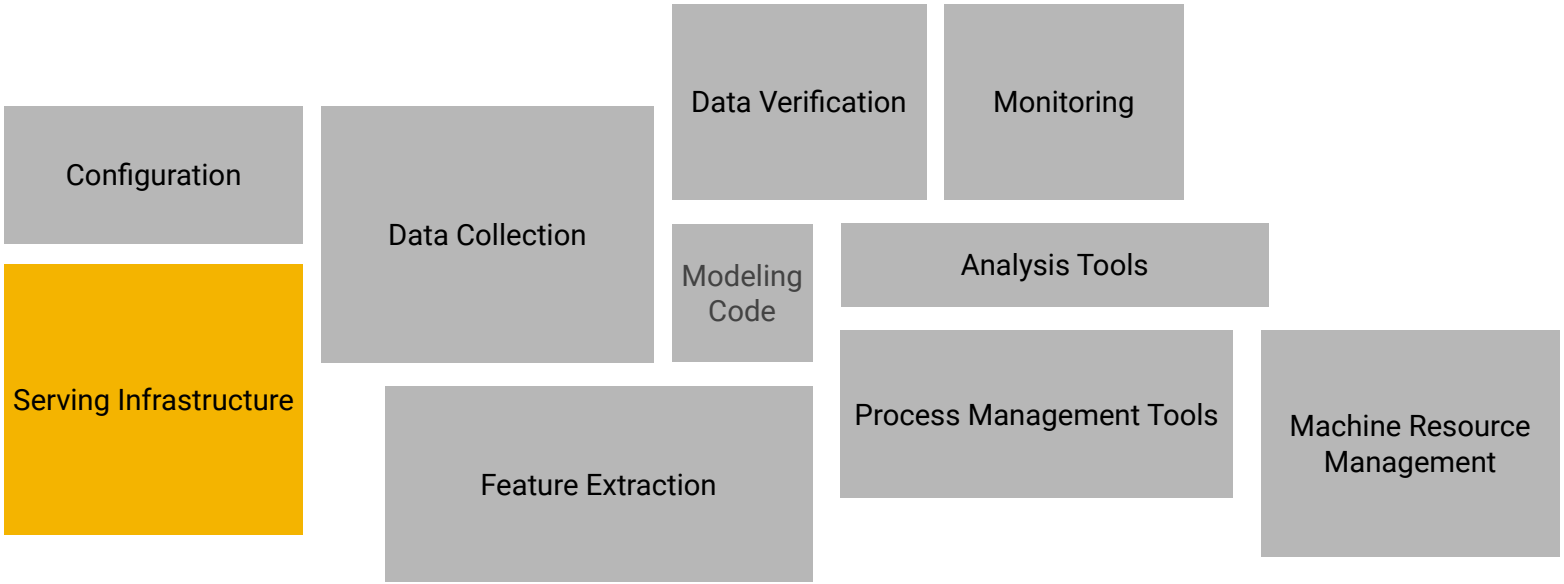

Client 1
Client 2
Client 3
TF Serving
Model v1


Client 1
Client 2
Client 3
TF Serving
Model v2


Client 1
Client 2
Client 3
TF Serving
Model v2


Client 1
Client 2
Client 3
Load Balancer
TF Serving
Model v2
TF Serving
Model v2
TF Serving
Model v2

TensorFlow Serving is part of TFX



Install TensorFlow Serving...

- Docker
- APT
- Build From Source
- PIP Packages

https://www.tensorflow.org/tfx/serving/setup

```
!echo "deb http://storage.googleapis.com/tensorflow-serving-apt stable tensorflow-model-server
tensorflow-model-server-universal"
      | tee /etc/apt/sources.list.d/tensorflow-serving.list && \

curl https://storage.googleapis.com/tensorflow-serving-apt/tensorflow-serving.release.pub.gpg
      | apt-key add -


!apt update


!apt-get install tensorflow-model-server
```

```
import tensorflow as tf
import numpy as np
from tensorflow import keras
model = tf.keras.Sequential([keras.layers.Dense(units=1, input_shape=[1])])
model.compile(optimizer='sgd', loss='mean_squared_error')

xs = np.array([-1.0,  0.0, 1.0, 2.0, 3.0, 4.0], dtype=float)
ys = np.array([-3.0, -1.0, 1.0, 3.0, 5.0, 7.0], dtype=float)
model.fit(xs, ys, epochs=500, verbose=2)

print(model.predict([10.0]))
```

```
tf.saved_model.simple_save(
    keras.backend.get_session(),
    export_path,
    inputs={'input_image': model.input},
    outputs={t.name:t for t in model.outputs})
```

```
tf.saved_model.simple_save(
    keras.backend.get_session(),
    export_path,
    inputs={'input_image': model.input},
    outputs={t.name:t for t in model.outputs})
```

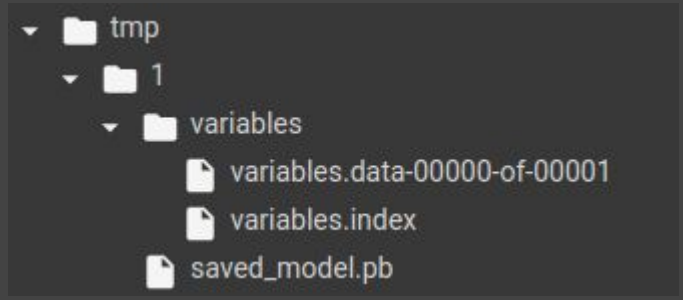
```
tf.saved_model.simple_save(
    keras.backend.get_session(),
    export_path,
    inputs={'input_image': model.input},
    outputs={t.name:t for t in model.outputs})
```

```
tf.saved_model.simple_save(
    keras.backend.get_session(),
    export_path,
    inputs={'input_image': model.input},
    outputs={t.name:t for t in model.outputs})
```

```
tf.saved_model.simple_save(
    keras.backend.get_session(),
    export_path,
    inputs={'input_image': model.input},
    outputs={t.name:t for t in model.outputs})
```

```
# Fetch the Keras session and save the model
# The signature definition is defined by the input and output tensors,
# and stored with the default serving key
import tempfile

MODEL_DIR = tempfile.gettempdir()
version = 1
export_path = os.path.join(MODEL_DIR, str(version))
print('export_path = {}\n'.format(export_path))
if os.path.isdir(export_path):
  print('\nAlready saved a model, cleaning up\n')
  !rm -r {export_path}

tf.saved_model.simple_save(
    keras.backend.get_session(),
    export_path,
    inputs={'input_image': model.input},
    outputs={t.name:t for t in model.outputs})

print('\nSaved model:')
!ls -l {export_path}
```

```
export_path = /tmp/1

Saved model:
total 44
-rw-r--r-- 1 root root 39532 Nov  3 23:35 saved_model.pb
drwxr-xr-x 2 root root  4096 Nov  3 23:35 variables
```



```
!saved_model_cli show --dir {export_path} --all
```

```
MetaGraphDef with tag-set: 'serve' contains the following SignatureDefs:

signature_def['serving_default']:
  The given SavedModel SignatureDef contains the following input(s):
    inputs['input_image'] tensor_info:
        dtype: DT_FLOAT
        shape: (-1, 1)
        name: dense_input:0
  The given SavedModel SignatureDef contains the following output(s):
    outputs['dense/BiasAdd:0'] tensor_info:
        dtype: DT_FLOAT
        shape: (-1, 1)
        name: dense/BiasAdd:0
  Method name is: tensorflow/serving/predict
```

```
MetaGraphDef with tag-set: 'serve' contains the following SignatureDefs:

signature_def['serving_default']:
  The given SavedModel SignatureDef contains the following input(s):
    inputs['input_image'] tensor_info:
        dtype: DT_FLOAT
        shape: (-1, 1)
        name: dense_input:0
  The given SavedModel SignatureDef contains the following output(s):
    outputs['dense/BiasAdd:0'] tensor_info:
        dtype: DT_FLOAT
        shape: (-1, 1)
        name: dense/BiasAdd:0
  Method name is: tensorflow/serving/predict
```

```
MetaGraphDef with tag-set: 'serve' contains the following SignatureDefs:

signature_def['serving_default']:
  The given SavedModel SignatureDef contains the following input(s):
    inputs['input_image'] tensor_info:
        dtype: DT_FLOAT
        shape: (-1, 1)
        name: dense_input:0
  The given SavedModel SignatureDef contains the following output(s):
    outputs['dense/BiasAdd:0'] tensor_info:
        dtype: DT_FLOAT
        shape: (-1, 1)
        name: dense/BiasAdd:0
  Method name is: tensorflow/serving/predict
```

```
MetaGraphDef with tag-set: 'serve' contains the following SignatureDefs:

signature_def['serving_default']:
  The given SavedModel SignatureDef contains the following input(s):
    inputs['input_image'] tensor_info:
        dtype: DT_FLOAT
        shape: (-1, 1)
        name: dense_input:0
  The given SavedModel SignatureDef contains the following output(s):
    outputs['dense/BiasAdd:0'] tensor_info:
        dtype: DT_FLOAT
        shape: (-1, 1)
        name: dense/BiasAdd:0
  Method name is: tensorflow/serving/predict
```

```
os.environ["MODEL_DIR"] = MODEL_DIR

%%bash --bg
nohup tensorflow_model_server \
  --rest_api_port=8501 \
  --model_name=helloworld \
  --model_base_path="${MODEL_DIR}" >server.log 2>&1
```

```
os.environ["MODEL_DIR"] = MODEL_DIR

%%bash --bg
nohup tensorflow_model_server \
  --rest_api_port=8501 \
  --model_name=helloworld \
  --model_base_path="${MODEL_DIR}" >server.log 2>&1
```

```
os.environ["MODEL_DIR"] = MODEL_DIR

%%bash --bg
nohup tensorflow_model_server \
  --rest_api_port=8501 \
  --model_name=helloworld \
  --model_base_path="${MODEL_DIR}" >server.log 2>&1
```

```
os.environ["MODEL_DIR"] = MODEL_DIR

%%bash --bg
nohup tensorflow_model_server \
  --rest_api_port=8501 \
  --model_name=helloworld \
  --model_base_path="${MODEL_DIR}" >server.log 2>&1
```

```
os.environ["MODEL_DIR"] = MODEL_DIR

%%bash --bg
nohup tensorflow_model_server \
  --rest_api_port=8501 \
  --model_name=helloworld \
  --model_base_path="${MODEL_DIR}" >server.log 2>&1
```

```
os.environ["MODEL_DIR"] = MODEL_DIR

%%bash --bg
nohup tensorflow_model_server \
  --rest_api_port=8501 \
  --model_name=helloworld \
  --model_base_path="${MODEL_DIR}" >server.log 2>&1
```

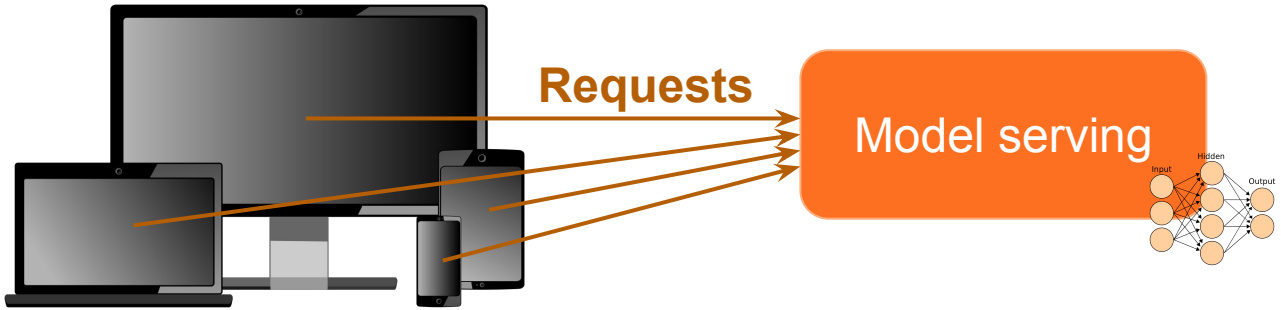
```
os.environ["MODEL_DIR"] = MODEL_DIR

%%bash --bg
nohup tensorflow_model_server \
  --rest_api_port=8501 \
  --model_name=helloworld \
  --model_base_path="${MODEL_DIR}" >server.log 2>&1
```

```
!tail server.log
```

```
2019-11-03 23:52:16.178660: I external/org_tensorflow/tensorflow/cc/saved_model/reader.cc:54] Reading meta graph with
tags { serve }
2019-11-03 23:52:16.179306: I external/org_tensorflow/tensorflow/core/platform/cpu_feature_guard.cc:142] Your CPU
supports instructions that this TensorFlow binary was not compiled to use: AVX2 FMA
2019-11-03 23:52:16.192356: I external/org_tensorflow/tensorflow/cc/saved_model/loader.cc:202] Restoring SavedModel
bundle.
2019-11-03 23:52:16.198559: I external/org_tensorflow/tensorflow/cc/saved_model/loader.cc:311] SavedModel load for
tags { serve }; Status: success. Took 20743 microseconds.
2019-11-03 23:52:16.198779: I tensorflow_serving/servables/tensorflow/saved_model_warmup.cc:105] No warmup data file
found at /tmp/1/assets.extra/tf_serving_warmup_requests
2019-11-03 23:52:16.198852: I tensorflow_serving/core/loader_harness.cc:87] Successfully loaded servable version
{name: helloworld version: 1}
2019-11-03 23:52:16.199907: I tensorflow_serving/model_servers/server.cc:353] Running gRPC ModelServer at
0.0.0.0:8500 ...
[warn] getaddrinfo: address family for nodename not supported
2019-11-03 23:52:16.200544: I tensorflow_serving/model_servers/server.cc:373] Exporting HTTP/REST API
at:localhost:8501 ...
[evhttp_server.cc : 238] NET_LOG: Entering the event loop ...
```

```
!tail server.log
```

```
2019-11-03 23:52:16.178660: I external/org_tensorflow/tensorflow/cc/saved_model/reader.cc:54] Reading meta graph with
tags { serve }
2019-11-03 23:52:16.179306: I external/org_tensorflow/tensorflow/core/platform/cpu_feature_guard.cc:142] Your CPU
supports instructions that this TensorFlow binary was not compiled to use: AVX2 FMA
2019-11-03 23:52:16.192356: I external/org_tensorflow/tensorflow/cc/saved_model/loader.cc:202] Restoring SavedModel
bundle.
2019-11-03 23:52:16.198559: I external/org_tensorflow/tensorflow/cc/saved_model/loader.cc:311] SavedModel load for
tags { serve }; Status: success. Took 20743 microseconds.
2019-11-03 23:52:16.198779: I tensorflow_serving/servables/tensorflow/saved_model_warmup.cc:105] No warmup data file
found at /tmp/1/assets.extra/tf_serving_warmup_requests
2019-11-03 23:52:16.198852: I tensorflow_serving/core/loader_harness.cc:87] Successfully loaded servable version
{name: helloworld version: 1}
2019-11-03 23:52:16.199907: I tensorflow_serving/model_servers/server.cc:353] Running gRPC ModelServer at
0.0.0.0:8500 ...
[warn] getaddrinfo: address family for nodename not supported
2019-11-03 23:52:16.200544: I tensorflow_serving/model_servers/server.cc:373] Exporting HTTP/REST API
at:localhost:8501 ...
[evhttp_server.cc : 238] NET_LOG: Entering the event loop ...
```


Requests
Model serving
Input
Hidden
Output


Predictions
Model serving
Input
Hidden
Output

```
import json
xs = np.array([[9.0], [10.0]])
data = json.dumps({"signature_name": "serving_default", "instances": xs.tolist()})
print(data)
```

```
import json
xs = np.array([[9.0], [10.0]])
data = json.dumps({"signature_name": "serving_default", "instances": xs.tolist()})
print(data)
```

```
import json
xs = np.array([[9.0], [10.0]])
data = json.dumps({"signature_name": "serving_default", "instances": xs.tolist()})
print(data)
```

```
import json
xs = np.array([[9.0], [10.0]])
data = json.dumps({"signature_name": "serving_default", "instances": xs.tolist()})
print(data)
```

```
import json
xs = np.array([[9.0], [10.0]])
data = json.dumps({"signature_name": "serving_default", "instances": xs.tolist()})
print(data)
```

```
import json
xs = np.array([[9.0], [10.0]])
data = json.dumps({"signature_name": "serving_default", "instances": xs.tolist()})
print(data)
```

```
MetaGraphDef with tag-set: 'serve' contains the following SignatureDefs:

signature_def['serving_default']:
  The given SavedModel SignatureDef contains the following input(s):
    inputs['input_image'] tensor_info:
        dtype: DT_FLOAT
        shape: (-1, 1)
        name: dense_input:0
  The given SavedModel SignatureDef contains the following output(s):
    outputs['dense/BiasAdd:0'] tensor_info:
        dtype: DT_FLOAT
        shape: (-1, 1)
        name: dense/BiasAdd:0
  Method name is: tensorflow/serving/predict
```

```
import json
xs = np.array([[9.0], [10.0]])
data = json.dumps({"signature_name": "serving_default", "instances": xs.tolist()})
print(data)
```

```
import json
xs = np.array([[9.0], [10.0]])
data = json.dumps({"signature_name": "serving_default", "instances": xs.tolist()})
print(data)
```

```
{"signature_name": "serving_default", "instances": [[9.0], [10.0]]}
```

```
!pip install -q requests

import requests
headers = {"content-type": "application/json"}
json_response = requests.post('http://localhost:8501/v1/models/helloworld:predict',
data=data, headers=headers)

print(json_response.text)
predictions = json.loads(json_response.text)['predictions']
```

```
!pip install -q requests

import requests

headers = {"content-type": "application/json"}

json_response = requests.post('http://localhost:8501/v1/models/helloworld:predict',
                              data=data,
                              headers=headers)

print(json_response.text)

predictions = json.loads(json_response.text)['predictions']
```

```
!pip install -q requests

import requests

headers = {"content-type": "application/json"}

json_response = requests.post('http://localhost:8501/v1/models/helloworld:predict',
                              data=data,
                              headers=headers)

print(json_response.text)

predictions = json.loads(json_response.text)['predictions']
```

```
!pip install -q requests

import requests

headers = {"content-type": "application/json"}

json_response = requests.post('http://localhost:8501/v1/models/helloworld:predict',
                              data=data,
                              headers=headers)

print(json_response.text)

predictions = json.loads(json_response.text)['predictions']
```

```
!pip install -q requests

import requests

headers = {"content-type": "application/json"}

json_response = requests.post('http://localhost:8501/v1/models/helloworld:predict',
                              data=data,
                              headers=headers)

print(json_response.text)

predictions = json.loads(json_response.text)['predictions']
```

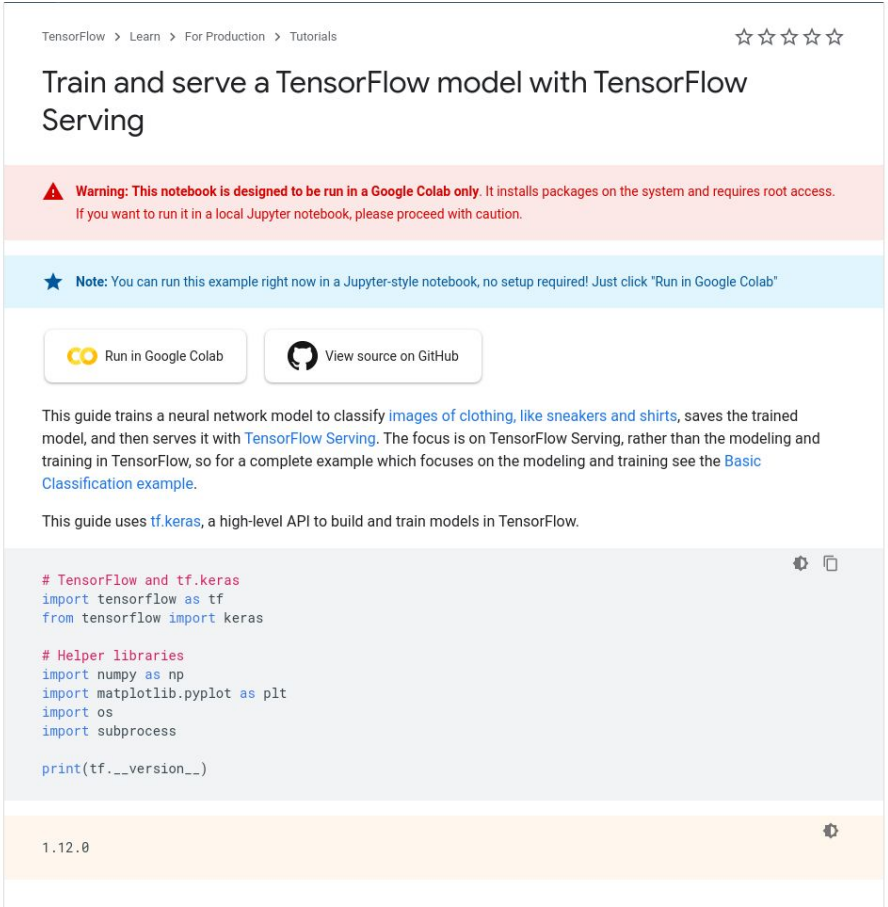
```
!pip install -q requests

import requests

headers = {"content-type": "application/json"}

json_response = requests.post('http://localhost:8501/v1/models/helloworld:predict',
                              data=data,
                              headers=headers)

print(json_response.text)

predictions = json.loads(json_response.text)['predictions']
```

```
!pip install -q requests

import requests

headers = {"content-type": "application/json"}

json_response = requests.post('http://localhost:8501/v1/models/helloworld:predict',
                              data=data,
                              headers=headers)

print(json_response.text)

predictions = json.loads(json_response.text)['predictions']
```

```
{
    "predictions": [[16.9865685], [18.984293]]
}
```

http://bit.ly/tfserving-lab1

https://www.tensorflow.org/tfx/serving/tutorials/Serving_REST_simple



TensorFlow > Learn > For Production > Tutorials

Train and serve a TensorFlow model with TensorFlow Serving

Warning: This notebook is designed to be run in a Google Colab only. It installs packages on the system and requires root access. If you want to run it in a local Jupyter notebook, please proceed with caution.

Note: You can run this example right now in a Jupyter-style notebook, no setup required! Just click "Run in Google Colab"

Run in Google Colab | View source on GitHub

This guide trains a neural network model to classify images of clothing, like sneakers and shirts, saves the trained model, and then serves it with TensorFlow Serving. The focus is on TensorFlow Serving, rather than the modeling and training in TensorFlow, so for a complete example which focuses on the modeling and training see the Basic Classification example.

This guide uses tf.keras, a high-level API to build and train models in TensorFlow.

```
# TensorFlow and tf.keras
import tensorflow as tf
from tensorflow import keras

# Helper libraries
import numpy as np
import matplotlib.pyplot as plt
import os
import subprocess

print(tf.__version__)
```

```
1.12.0
```

```
import json

data = json.dumps({"signature_name": "serving_default",
                   "instances": test_images[0:3].tolist()})

print('Data: {} ... {}'.format(data[:50], data[len(data)-52:]))


print(data)
```

{"instances": [
[[[0.0], [0.0]],
[[0.0], [0.0]],
[[0.0], [0.0]],
[[0.0], [0.0]],
[[0.0], [0.0]],
[[0.0], [0.0]],
[[0.0], [0.0]],
[[0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.011764705882352941], [0.00392156862745098], [0.0], [0.0], [0.027450980392156862], [0.0], [0.14509803921
[[0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.00392156862745098], [0.00784313725490196], [0.0], [0.10588235294117647], [0.32941176470588235], [0.043137254901960784], [0.0], [0.0], [0.0], [0.0
[[0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.00392156862745098], [0.0], [0.0], [0.34509803921568627], [0.5607843137254902], [0.43137254901960786], [0.0], [0.0], [0.0], [0.0], [0.086274509803
[[0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.01568627450980392], [0.0], [0.20784313725490197], [0.5058823529411764], [0.47058823529411764], [0.5764705882352941], [0.6862745098039216], [0.615
[[0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.00784313725490196], [0.0], [0.043137254901960784], [0.5372549019607843], [0.5098039215686274], [0.5019607843137255], [0.6274509803921569], [0.6901960784
[[0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.00392156862745098], [0.0], [0.00784313725490196], [0.00392156862745098], [0.0], [0.011764705882352941], [0.0], [0.0], [0.45098039215686275], [0.4470588235294118], [0.41568627450980394], [0.53725
[[0.0], [0.0], [0.0], [0.0], [0.00392156862745098], [0.0], [0.0], [0.0], [0.0], [0.0], [0.011764705882352941], [0.0], [0.0], [0.34901960784313724], [0.5450980392156862], [0.35294117647058826], [0.3686274509803922], [0.6], [0.584313725490196
[[0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.00784313725490196], [0.01568627450980392], [0.00392156862745098], [0.0], [0.0], [0.0], [0.3843137254901961], [0.5333333333333333], [0.43137254901960786], [0.42745098039215684], [0.43137254901960
[[0.0], [0.0], [0.00784313725490196], [0.00784313725490196], [0.00392156862745098], [0.00784313725490196], [0.0], [0.0], [0.0], [0.0], [0.10196078431372549], [0.4235294117647059], [0.4588235294117647], [0.38823529411764707], [0.435294117647
[[0.011764705882352941], [0.0], [0.0], [0.0], [0.0], [0.0], [0.08235294117647059], [0.20784313725490197], [0.3607843137254902], [0.4588235294117647], [0.43529411764705883], [0.403921568627451], [0.45098039215686275], [0.5058823529411
[[0.0], [0.0], [0.09019607843137255], [0.21176470588235294], [0.2549019607843137], [0.2980392156862745], [0.3333333333333333], [0.4627450980392157], [0.5019607843137255], [0.4823529411764706], [0.43529411764705883], [0.44313725490196076], [
[[0.0], [0.26666666666666666], [0.3686274509803922], [0.35294117647058826], [0.43529411764705883], [0.4470588235294118], [0.43529411764705883], [0.4470588235294118], [0.45098039215686275], [0.4980392156862745], [0.5294117647058824], [0.5333
[[0.27450980392156865], [0.6627450980392157], [0.5058823529411764], [0.40784313725490196], [0.3843137254901961], [0.39215686274509803], [0.3686274509803922], [0.3803921568627451], [0.3843137254901961], [0.4], [0.4235294117647059], [0.415686
[[0.06274509803921569], [0.49411764705882355], [0.6705882352941176], [0.7372549019607844], [0.7372549019607844], [0.7215686274509804], [0.6705882352941176], [0.6], [0.5294117647058824], [0.47058823529411764], [0.49411764705882355], [0.49803
[[0.0], [0.0], [0.0], [0.047058823529411764], [0.2627450980392157], [0.41568627450980394], [0.6431372549019608], [0.7254901960784313], [0.7803921568627451], [0.8235294117647058], [0.8274509803921568], [0.8235294117647058], [0.81568627450980
[[0.0], [0.0]],
[[0.0], [0.0]],
[[0.0], [0.0]],
[[0.0], [0.0]],
[[0.0], [0.0]],
[[0.0], [0.0]]],

[[[0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.050980392156862744], [0.2627450980392157], [0.0], [0.0], [0.0], [0.0], [0.19607843137254902], [0.14901960784313725], [0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.0],
[[0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.03137254901960784], [0.47058823529411764], [0.8196078431372549], [0.8862745098039215], [0.9686274509803922], [0.9294117647058824], [1.0], [1.0], [1.0], [0.9686274509803922], [0.9333333333
[[0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.5372549019607843], [0.9372549019607843], [0.9882352941176471], [0.9529411764705882], [0.9176470588235294], [0.8980392156862745], [0.9333333333333333], [0.9568627450980393], [0.9647058823529412],
[[0.0], [0.0], [0.0], [0.0], [0.0], [0.4], [1.0], [0.9058823529411765], [0.8941176470588236], [0.8901960784313725], [0.8941176470588236], [0.9137254901960784], [0.9019607843137255], [0.9019607843137255], [0.8980392156862745], [0.89411764705
[[0.0], [0.0], [0.0], [0.0], [0.0], [0.9137254901960784], [0.9450980392156862], [0.8980392156862745], [0.9058823529411765], [1.0], [1.0], [0.9333333333333333], [0.9058823529411765], [0.8901960784313725], [0.9333333333333333], [0.96470588235
[[0.0], [0.0], [0.0], [0.0], [0.0], [0.9725490196078431], [0.9450980392156862], [0.9058823529411765], [1.0], [0.5843137254901961], [0.1843137254901961], [0.9882352941176471], [0.8941176470588236], [1.0], [0.9490196078431372], [0.84705882352
[[0.0], [0.0], [0.0], [0.0], [0.0], [1.0], [0.9411764705882353], [0.9098039215686274], [1.0], [0.058823529411764705], [0.0], [1.0], [0.9294117647058824], [0.7490196078431373], [0.0], [0.0], [0.8392156862745098], [1.0], [0.050980392156862744
[[0.0], [0.0], [0.0], [0.0], [0.023529411764705882], [1.0], [0.9333333333333333], [0.9372549019607843], [1.0], [0.6941176470588235], [0.0], [1.0], [1.0], [0.0], [0.5098039215686274], [0.4549019607843137], [0.1843137254901961], [0.2549019607
[[0.0], [0.0], [0.0], [0.0], [0.12549019607843137], [1.0], [0.9254901960784314], [0.9607843137254902], [1.0], [0.8], [0.0], [1.0], [0.32941176470588235], [0.0], [0.1450980392156863], [0.10980392156862745], [0.12156862745098039], [0.0], [0.0
[[0.0], [0.0], [0.0], [0.0], [0.20784313725490197], [1.0], [0.9254901960784314], [0.9803921568627451], [0.9803921568627451], [0.9058823529411765], [0.00784313725490196], [1.0], [0.08235294117647059], [0.0], [0.8666666666666667], [1.0], [0.9
[[0.0], [0.0], [0.0], [0.0], [0.3137254901960784], [1.0], [0.9294117647058824], [0.9803921568627451], [0.9411764705882353], [1.0], [0.0], [0.0], [0.15294117647058825], [0.615686274509804], [0.0], [0.0], [0.8431372549019608], [0.368627450980
[[0.0], [0.0], [0.0], [0.0], [0.396078431372549], [1.0], [0.9215686274509803], [0.9921568627450981], [0.9568627450980393], [0.9529411764705882], [0.5215686274509804], [0.5411764705882353], [0.8156862745098039], [1.0], [0.788235294117647], [
[[0.0], [0.0], [0.0], [0.0], [0.49411764705882355], [1.0], [0.9137254901960784], [1.0], [0.9725490196078431], [0.9137254901960784], [1.0], [1.0], [0.9411764705882353], [0.9098039215686274], [0.9529411764705882], [0.9529411764705882], [0.905
[[0.0], [0.0], [0.0], [0.0], [0.5764705882352941], [1.0], [0.9137254901960784], [0.9764705882352941], [0.7098039215686275], [0.9529411764705882], [0.8901960784313725], [0.8784313725490196], [0.9019607843137255], [0.9176470588235294], [0.901
[[0.0], [0.0], [0.0], [0.0], [0.6392156862745098], [1.0], [0.9607843137254902], [0.8666666666666667], [0.33725490196078434], [1.0], [0.9137254901960784], [0.9137254901960784], [0.9215686274509803], [0.9254901960784314], [0.9176470588235294]
[[0.0], [0.0], [0.0], [0.0], [0.7098039215686275], [0.996078431372549], [1.0], [0.7843137254901961], [0.27058823529411763], [1.0], [0.8941176470588236], [0.9098039215686274], [0.9176470588235294], [0.9215686274509803], [0.9176470588235294],
[[0.0], [0.0], [0.0], [0.0], [0.7725490196078432], [0.9686274509803922], [1.0], [0.7372549019607844], [0.43137254901960786], [1.0], [0.8784313725490196], [0.9137254901960784], [0.9176470588235294], [0.9176470588235294], [0.9176470588235294]
[[0.0], [0.0], [0.0], [0.0], [0.7843137254901961], [0.9647058823529412], [1.0], [0.5843137254901961], [0.5686274509803921], [1.0], [0.8745098039215686], [0.9215686274509803], [0.9176470588235294], [0.9215686274509803], [0.9215686274509803],
[[0.0], [0.0], [0.0], [0.0], [0.8], [0.9529411764705882], [1.0], [0.43529411764705883], [0.6784313725490196], [1.0], [0.8901960784313725], [0.9215686274509803], [0.9215686274509803], [0.9254901960784314], [0.9215686274509803], [0.9215686274
[[0.0], [0.0], [0.0], [0.0], [0.7686274509803922], [0.9411764705882353], [1.0], [0.42745098039215684], [0.8352941176470589], [0.9803921568627451], [0.8980392156862745], [0.9215686274509803], [0.9215686274509803], [0.9254901960784314], [0.92
[[0.0], [0.0], [0.0], [0.0], [0.7529411764705882], [0.9529411764705882], [1.0], [0.4470588235294118], [0.9098039215686274], [0.9411764705882353], [0.9098039215686274], [0.9215686274509803], [0.9215686274509803], [0.9254901960784314], [0.917
[[0.0], [0.0], [0.0], [0.0], [0.7411764705882353], [0.984313725490196], [1.0], [0.6039215686274509], [0.9333333333333333], [0.9137254901960784], [0.9254901960784314], [0.9176470588235294], [0.9215686274509803], [0.9254901960784314], [0.9215
[[0.0], [0.0], [0.0], [0.0], [0.7176470588235294], [0.9882352941176471], [1.0], [0.6705882352941176], [0.9686274509803922], [0.9098039215686274], [0.9176470588235294], [0.9176470588235294], [0.9137254901960784], [0.9137254901960784], [0.909
[[0.0], [0.0], [0.0], [0.0], [0.6980392156862745], [0.9529411764705882], [1.0], [0.2235294117647059], [0.9333333333333333], [0.9450980392156862], [0.9333333333333333], [0.9333333333333333], [0.9333333333333333], [0.9294117647058824], [0.925
[[0.0], [0.0], [0.0], [0.0], [0.7372549019607844], [0.9411764705882353], [0.9803921568627451], [0.24313725490196078], [0.8549019607843137], [1.0], [0.8627450980392157], [0.8705882352941177], [0.8705882352941177], [0.8705882352941177], [0.87
[[0.0], [0.0], [0.0], [0.0], [0.5098039215686274], [0.9607843137254902], [0.9490196078431372], [0.09411764705882353], [0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.13333333333333333], [0.94901
[[0.0], [0.0], [0.0], [0.0], [0.2980392156862745], [1.0], [0.9764705882352941], [0.08627450980392157], [0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.15294117647058825], [0.9764705882352941], [
[[0.0], [0.0], [0.0], [0.0], [0.19215686274509805], [0.803921568627451], [0.7725490196078432], [0.043137254901960784], [0.0], [0.01568627450980392], [0.00392156862745098], [0.00784313725490196], [0.00784313725490196], [0.00784313725490196],

[[[0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.00392156862745098], [0.0], [0.2627450980392157], [0.6941176470588235], [0.5058823529411764], [0.6], [0.4588235294117647], [0.5058823529411764], [0.5725490196078431], [0.5529411764
[[0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.00784313725490196], [0.0], [0.7686274509803922], [1.0], [1.0], [1.0], [0.9450980392156862], [0.984313725490196], [1.0], [0.9607843137254902], [1.0], [0.2980392156862745], [0.0], [0
[[0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.0], [0.9529411764705882], [0.9294117647058824], [0.8509803921568627], [0.8941176470588236], [0.9058823529411765], [0.8705882352941177], [0.8549019607843137], [0.858823529411

```
[[[0.0], [0.0], [0.0], ... ]],
[[0.0], [0.0], [0.0], ... ]],
[[0.0], [0.0], [0.0], ...]]],


[[[0.0], [0.0], [0.0], ... ]],
[[0.0], [0.0], [0.0], ... ]],
[[0.0], [0.0], [0.0], ...]]],


[[[0.0], [0.0], [0.0], ... ]],
[[0.0], [0.0], [0.0], ... ]],
[[0.0], [0.0], [0.0], ...]]]
```

```
[[[0.0], [0.0], [0.0], ... ]],
[[0.0], [0.0], [0.0], ... ]],
[[0.0], [0.0], [0.0], ...]]],


[[[0.0], [0.0], [0.0], ... ]],
[[0.0], [0.0], [0.0], ... ]],
[[0.0], [0.0], [0.0], ...]]],


[[[0.0], [0.0], [0.0], ... ]],
[[0.0], [0.0], [0.0], ... ]],
[[0.0], [0.0], [0.0], ...]]]
```

```
[[[0.0], [0.0], [0.0], ... ]],
[[0.0], [0.0], [0.0], ... ]],
[[0.0], [0.0], [0.0], ...]]],


[[[0.0], [0.0], [0.0], ... ]],
[[0.0], [0.0], [0.0], ... ]],
[[0.0], [0.0], [0.0], ...]]],


[[[0.0], [0.0], [0.0], ... ]],
[[0.0], [0.0], [0.0], ... ]],
[[0.0], [0.0], [0.0], ...]]]
```

```
!pip install -q requests

import requests

headers = {"content-type": "application/json"}

json_response = requests.post('http://localhost:8501/v1/models/fashion_model:predict',
                              data=data, headers=headers)

predictions = json.loads(json_response.text)['predictions']
```

```
[
[5.77123615e-07, 2.66907847e-08, 4.7217938e-08, 1.97792871e-09, 5.31984341e-08,
0.00734644197, 3.1462946e-07, 0.0439051725, 0.000500570168, 0.948246837],

[0.00227244, 6.12080342e-09, 0.967876315, 3.0579281e-06, 0.0183339939, 3.18483538e-11,
0.011510049, 1.38639566e-14, 4.19033222e-06, 4.40264526e-11],

[1.45221502e-05, 0.999841571, 3.96758715e-08, 0.000131023204, 1.22008023e-05,
1.18227668e-08, 5.97860179e-08, 1.31281848e-08, 5.49047854e-07, 2.97885189e-10]
]
```

```
[
[5.77123615e-07, 2.66907847e-08, 4.7217938e-08, 1.97792871e-09, 5.31984341e-08,
0.00734644197, 3.1462946e-07, 0.0439051725, 0.000500570168, 0.948246837],

[0.00227244, 6.12080342e-09, 0.967876315, 3.0579281e-06, 0.0183339939, 3.18483538e-11,
0.011510049, 1.38639566e-14, 4.19033222e-06, 4.40264526e-11],

[1.45221502e-05, 0.999841571, 3.96758715e-08, 0.000131023204, 1.22008023e-05,
1.18227668e-08, 5.97860179e-08, 1.31281848e-08, 5.49047854e-07, 2.97885189e-10]
]
```

```
show(0,
'The model saw {} (class {}), and it was actually a {} (class {})'.format(
      class_names[np.argmax(predictions[0])], test_labels[0],
      class_names[np.argmax(predictions[0])], test_labels[0]))
```

The model thought this was a Ankle boot (class 9), and it was actually a Ankle boot (class 9)