All scripts (talking heads, console sessions etc.) can be found in a single google doc

https://docs.google.com/document/d/1r4imjXeFWAjbGt1jKdwsC
usp=sharing

Link should allow for editing / commenting

Slides:

https://djcordhose.github.io/ai/course_u4.html

Slides ready for PDF Printing

https://djcordhose.github.io/ai/course_u4.html?print-pdf

PDF:

U4-M1-INTRO

Talking Head

U4-M2: DEPLOYING TO THE BROWSER

ML Car Insurance Risk Calculator

https://djcordhose.github.io/deep-learning-crash-course-notebooks/

A JavaScript library for training and deploying ML models in the browser and on Node.js

Develop ML with JavaScript

Use flexible and intuitive APIs to build and train models from scratch using the low-level JavaScript linear algebra library or the high-level layers API

Run Existing Models

Use TensorFlow.js model converters to run pre-existing TensorFlow models right in the browser or under Node.js.

Retrain Existing Models

Retrain pre-existing ML models using sensor data connected to the browser, or other client-side data.

https://js.tensorflow.org/

U4-M3: Converting our Keras model for tensorflow.js

https://colab.research.google.com/github/djcordhose/deep-learning-crash-course-notebooks/blob/master/U4-M3-tensorflowjs.ipynb

Converting our Keras Model to tensorflow.js

```
tensorflowjs_converter --input_format keras \
./model/insurance.hdf5 \
./tfjs
```

https://js.tensorflow.org/tutorials/import-keras.html

Loading and using directly from the Browser

```
const model = await tf.loadModel('tfjs/model.json');
```

```
// max speed, age, thousand miles per year
const example = tf.tensor([[100, 47, 10]]);
const prediction = model.predict(example);
console.log(await prediction.data());
//[0.003348011290654, 0.871034324169158, 0.12561771273612976]
```

https://djcordhose.github.io/deep-learning-crash-course-notebooks/load_model.html

U4-M3: JAVASCRIPT GLUE CODE FOR FINAL APPLICATION

Code in IDE

Exercise

Install the Risk Calculator locally and change it to use your data

Exercise

Install the Risk Calculator locally and change it to use your data

- clone or download https://github.com/DJCordhose/deep-learning-crash-course-notebooks

Exercise

Install the Risk Calculator locally and change it to use your data

- clone or download https://github.com/DJCordhose/deep-learning-crash-course-notebooks
- in the ***docs*** folder you find the complete Risk Calculator

Exercise

Install the Risk Calculator locally and change it to use your data

- clone or download https://github.com/DJCordhose/deep-learning-crash-course-notebooks
- in the ***docs*** folder you find the complete Risk Calculator
- run a local web server in that directory

Exercise

Install the Risk Calculator locally and change it to use your data

- clone or download https://github.com/DJCordhose/deep-learning-crash-course-notebooks
- in the ***docs*** folder you find the complete Risk Calculator
- run a local web server in that directory
  - if you do not have one you can use https://www.npmjs.com/package/http-server

Exercise

Install the Risk Calculator locally and change it to use your data

- clone or download https://github.com/DJCordhose/deep-learning-crash-course-notebooks
- in the ***docs*** folder you find the complete Risk Calculator
- run a local web server in that directory
  - if you do not have one you can use https://www.npmjs.com/package/http-server
- in your favorite editor or IDE open ***index.html***

Exercise

Install the Risk Calculator locally and change it to use your data

- clone or download https://github.com/DJCordhose/deep-learning-crash-course-notebooks
- in the ***docs*** folder you find the complete Risk Calculator
- run a local web server in that directory
  - if you do not have one you can use https://www.npmjs.com/package/http-server
- in your favorite editor or IDE open ***index.html***
- find the line where I entered my own personal data

Exercise

Install the Risk Calculator locally and change it to use your data

- clone or download https://github.com/DJCordhose/deep-learning-crash-course-notebooks
- in the ***docs*** folder you find the complete Risk Calculator
- run a local web server in that directory
  - if you do not have one you can use https://www.npmjs.com/package/http-server
- in your favorite editor or IDE open ***index.html***
- find the line where I entered my own personal data
- change it to yours and try it in the browser

U4-M4: Preparation for serving

https://colab.research.google.com/github/djcordhose/deep-learning-crash-course-notebooks/blob/master/U4-M4-tf-prep.ipynb

Deploying to Google Cloud ML

TALKING HEAD

Deploying to Google Cloud ML

- Takes away the scaling and installation burden

TALKING HEAD

Deploying to Google Cloud ML

- Takes away the scaling and installation burden
- Requires to convert your Keras model to a TensorFlow model

TALKING HEAD

Deploying to Google Cloud ML

- Takes away the scaling and installation burden
- Requires to convert your Keras model to a TensorFlow model
- You need to define Signatures for input and output

TALKING HEAD

Deploying to Google Cloud ML

- Takes away the scaling and installation burden
- Requires to convert your Keras model to a TensorFlow model
- You need to define Signatures for input and output
- The generated serving model also works for your local model servers (more on that later)

TALKING HEAD

Deploying to Google Cloud ML

- Takes away the scaling and installation burden
- Requires to convert your Keras model to a TensorFlow model
- You need to define Signatures for input and output
- The generated serving model also works for your local model servers (more on that later)
- Can use CPU or GPU

TALKING HEAD

Creating the Signature

```
signature = saved_model.signature_def_utils.build_signature_def(
    inputs={'inputs': build_tensor_info(model.input)},
    outputs={'scores': build_tensor_info(model.output)},
    method_name=saved_model.signature_constants.PREDICT_METHOD_NAME)
```

https://www.tensorflow.org/serving/signature_defs

Creating the builder and save model

```
builder = saved_model.builder.SavedModelBuilder("tf/1") # path to model
```

```
builder.add_meta_graph_and_variables(
    sess, [saved_model.tag_constants.SERVING],
    signature_def_map={
    saved_model.signature_constants.
        DEFAULT_SERVING_SIGNATURE_DEF_KEY: signature
    })
```

```
builder.save()
```

U4-M5: CHECKING OUR TF MODEL

https://colab.research.google.com/github/djcordhose/deep-learning-crash-course-notebooks/blob/master/U4-M5-tf-check.ipynb

Console Session (already recorded)

U4-M6: Hosting your model on Google Cloud ML

https://colab.research.google.com/github/djcordhose/deep-learning-crash-course-notebooks/blob/master/U4-M6-cloud.ipynb

Deploying to Google Clound ML

Copy your model to a Cloud Bucket

```
gsutil mb gs://my_bucket
gsutil cp -R tf/1 gs://my_bucket
```

Needs Google Cloud SDK: https://cloud.google.com/sdk/install

Deploy from this bucket

```
gcloud ml-engine models create "ml_insurance" --enable-logging
gcloud ml-engine versions create "v1" --model "ml_insurance" \
 --origin "gs://my_bucket/1"
gcloud ml-engine versions describe "v1" --model "ml_insurance"
```

https://cloud.google.com/ml-engine/docs/tensorflow/deploying-models
https://cloud.google.com/ml-engine/docs/tensorflow/prediction-overview#prediction_logging

Making Predictions

Input format is a bit special

```
# sample_insurance.json
{"inputs": [ 160,  18,  100]}
{"inputs": [ 100,  47,  10]}
{"inputs": [ 90,  20,  20]}
```

Call from Google Cloud Console

```
gcloud ml-engine predict --model "ml_insurance" --version "v1" \
 --json-instances ./sample_insurance.json
```

```
SCORES
[0.8658562898635864, 7.3186689185118e-14, 0.13414366543293]
[0.00276080076582729 8, 0.872088074684143 1, 0.12515118718147278]
[5.4529344197362 66e-05, 0.005952719133347273, 0.9939927458763123]
```

https://cloud.google.com/ml-engine/docs/tensorflow/online-predict

U4-M7: Running on a dedicated Linux server

https://colab.research.google.com/github/djcordhose/deep-learning-crash-course-notebooks/blob/master/U4-M7-local.ipynb

TensorFlow Serving REST API

Starting the Model Server in Rest Mode (needs Linux)

```
tensorflow_model_server --rest_api_port=8501 \
--model_name=manning_insurance_1 \
--model_base_path=$(pwd)/tf
```

https://www.tensorflow.org/serving/api_rest

Curling to it

```
curl -X POST \
http://localhost:8501/v1/models/manning_insurance_1:predict  \
-d '{ "instances": [{"inputs": [ 100.0,  47.0,  10.0]}]}'
# {
#     "predictions": [[0.0027608, 0.872088, 0.125151]]
# }
```

U4-M8: OUTRO / SUMMARY

Talking Head