## **Archaeology**

The following sections are a stroll back in history.

- History is important: it's hard to know how far we have gotten without seeing how far back we were
- You will encounter "history" when you bump into other people's notebooks (e.g., blog posts) -It's important to be able to distinguish between that which is obsolete and that which is current

Keras is now tightly integrated into TensorFlow (even more so in TensorFlow 2.0)

This cleans up a rather unruly TensorFlow eco-system that resulted in similar functionality in multiple places.

This can make it very confusing for someone new to TensorFlow.

There are lots of examples on the web written using various similar-looking packages.

I'll try to point out potential sources of confusion. Beware!

<u>Demystify the TensorFlow APIs (https://medium.com/google-developer-experts/demystify-the-tensorflow-apis-57d2b0b8b6c0)</u> summarizes it well

- tf.layers is going away in TensorFlow 2.0
  - tf.keras is recommended going forward
  - Do not use
- <u>Estimators (https://www.tensorflow.org/guide/estimators)</u> (tf.Estimator)
  - Estimators are sometimes called "models in a box"; somewhat similar to sklearn
    - pre-canned high-level models (like Classifiers) rather than low-level tf.keras.layers (like Dense) from which it is built
    - convenient interface to <u>Datasets for Estimators</u>
       (<a href="https://www.tensorflow.org/guide/datasets">https://www.tensorflow.org/guide/datasets</a> for estimators)
      - no need to create own mini-batches, etc.
  - You can achieve quite a bit of the convenience using Keras, so we will skip Estimators.

- Low-level TensorFlow
  - great for learning
  - better to rely on pre-defined layers when possible

#### And our own observations

- tf.contrib
  - this was a name-space created to enable users to contribute useful packages.
  - some of these packages may have made their way into the core, or been integrated elsewhere
    - tf.contrib.learn.Estimator is the obsolete version of tf.Estimator
  - eliminated from TensorFlow 2.0
    - avoid
- <u>Datasets API (https://www.tensorflow.org/guide/datasets)</u>
  - an API to handle large datasets, in memory-

We will focus on two styles or packages in our course

- tf.keras
  - this is the future, as it will be tightly integrated into TensorFlow 2.0
- tf.layers modules (e.g., tf.layers.dense)
  - used only to be compatible with the Geron book.
  - it is slightly lower level than Keras

## tensorflow.keras vs keras (Confusion alert)

```
TL;DR
```

#### YES

```
import tensorflow as tf
tf.keras.layers.Dense(...)
```

 from tensorflow import keras keras.layers.Dense(...)

#### NO

import keras keras.layers.Dense( ... ) Technically speaking: Keras is an API -- a specification -- not a library.

- TensorFlow has implemented this specification as a submodule of the TensorFlow module:
  - tensorflow.keras
- There is a separate Keras project and module: keras
  - that supports multiple "backends", including TensorFlow
  - Cannot run Python versions > 3.6 (one backend isn't cooperating)

#### This is not just a legal difference

they are separate modules that do very similar things

#### This may get confusing

- The <u>TensorFlow docs for Keras (https://www.tensorflow.org/guide/keras)</u> refers to TensorFlow's implementation of the API
  - used as from tensorflow import keras
  - this is what we will use!
  - other syntactic forms to use: tf.keras...
- The <u>Keras docs (https://keras.io/)</u> refers to the abstract Keras API and keras module
  - used as import keras as keras

### tensorflow.keras

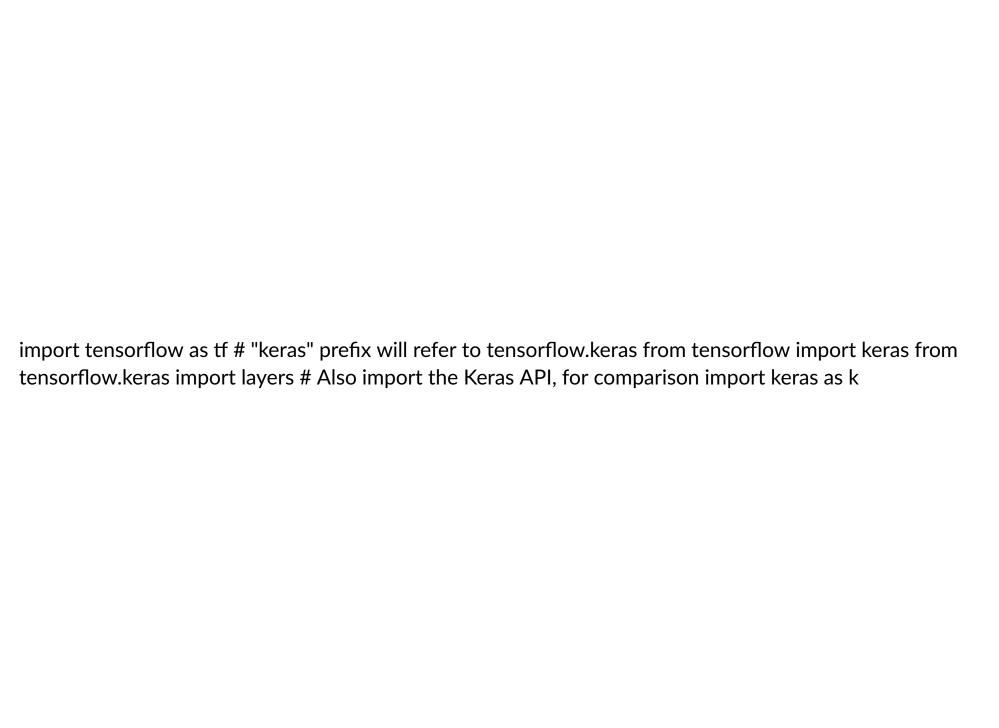
<u>Guidance from TensorFlow team (https://medium.com/tensorflow/standardizing-on-keras-guidance-on-high-level-apis-in-tensorflow-2-0-bad2b04c819a)</u>

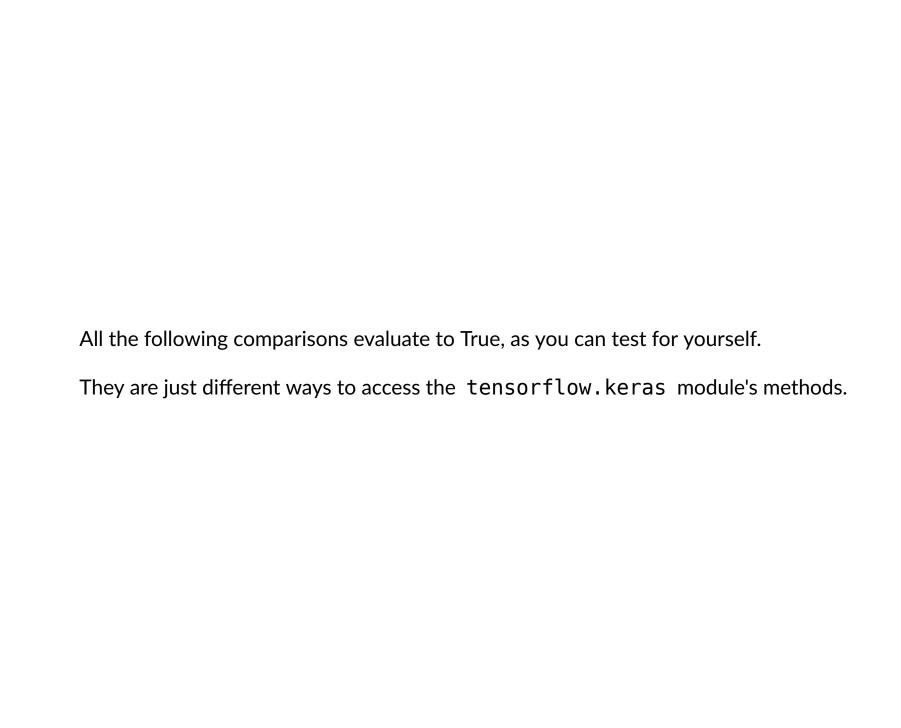
- tf.keras is an implementation of the Keras API
  - with enhancements
    - eager execution
  - integrated into TensorFlow ecosystem
    - ∘ tf.data

import tensorflow as tf Dense = tf.keras.layers.Dense model = tf.keras.Sequential() model.add(layers.Dense(64, activation='relu'))

# Technical point: showing the difference between tensorflow.keras and keras

Here we demonstrate that although the two modules implement the same methods, the are different methods





tf.keras.layers.Dense == keras.layers.Dense tf.keras.layers.Dense == layers.Dense

But the following is **not** True because they come from different packages:

- one from tensorflow.keras
- one from keras

tf.keras.layers.Dense == k.layers.Dense

```
In [1]: print("Done")
```

Done