1. **How would you describe TensorFlow in a short sentence?**

TensorFlow is an open source framework developed by Google researchers to run machine learning, deep learning and other statistical and predictive analytics workloads.

**What are its main features?**

* Language Support.
* Drag and Drop.
* Pre-Built Algorithms.
* Model Training.
* Pre-Built Algorithms.
* Model Training.
* Feature Engineering.

**Can you name other popular Deep Learning libraries?**

* Theano, deeplearning4j, Torch, and Caffe...

1. **Is TensorFlow a drop-in replacement for NumPy? What are the main differences between the two?**

* NumPy and TensorFlow are actually very similar in many respects. Both are, essentially, array manipulation libraries, built around the concept of tensors (or nd-arrays, in NumPy terms), the key difference between tensors and NumPy arrays is that tensors have accelerator support like GPU and TPU and are immutable.

1. **Do you get the same result with tf.range(10) and tf.constant(np.arange(10))?**

* tf.range is not equivalent to np.arange

1. **Can you name six other data structures available in TensorFlow, beyond regular tensors?**

* TensorFlow accepts Python native types like **booleans, strings and numeric (int, float)**.

1. **A custom loss function can be defined by writing a function or by subclassing the keras.losses.Loss class. When would you use each option?**

* This option can then be passed at the **compile stage.**

1. **Similarly, a custom metric can be defined in a function or a subclass of keras.metrics.Metric. When would you use each option?**
2. **When should you create a custom layer versus a custom model?**

* Most models are made of layers. Layers are **functions with a known mathematical structure that can be reused(custom layer) and have trainable variables**.

1. **What are some use cases that require writing your own custom training loop?**

* Write for your training audience.
* Write to your training audience.
* Use conversational language for training

1. Can custom Keras components contain arbitrary Python code, or must they be convertible to TF Functions?

* tf. function **takes a regular function as input and returns a Function** . A Function is a Python callable that builds TensorFlow graphs from the Python function. You use a Function in the same way as its Python equivalent.

1. **What are the main rules to respect if you want a function to be convertible to a TF Function?**

**-**

1. **When would you need to create a dynamic Keras model? How do you do that? Why not make all your models dynamic?**