tf.train.natural_exp_decay

```
natural_exp_decay(
    learning_rate,
    global_step,
    decay_steps,
    decay_rate,
    staircase=False,
    name=None
)
```

Defined in `tensorflow/python/training/learning_rate_decay.py`.

See the guide: Training > Decaying the learning rate

Applies natural exponential decay to the initial learning rate.

When training a model, it is often recommended to lower the learning rate as the training progresses. This function applies an exponential decay function to a provided initial learning rate. It requires an `global_step` value to compute the decayed learning rate. You can just pass a TensorFlow variable that you increment at each training step.

The function returns the decayed learning rate. It is computed as:

```
decayed_learning_rate = learning_rate * exp(-decay_rate * global_step)
```

Example: decay exponentially with a base of 0.96:

```
...
global_step = tf.Variable(0, trainable=False)
learning_rate = 0.1
k = 0.5
learning_rate = tf.train.exponential_time_decay(learning_rate, global_step, k)

# Passing global_step to minimize() will increment it at each step.
learning_step = (
    tf.train.GradientDescentOptimizer(learning_rate)
    .minimize(...my loss..., global_step=global_step)
)
```

Args:

- `learning_rate`: A scalar `float32` or `float64` `Tensor` or a Python number. The initial learning rate.
- `global_step`: A Python number. Global step to use for the decay computation. Must not be negative.
- `decay_steps`: How often to apply decay.
- `decay_rate`: A Python number. The decay rate.
- `staircase`: Whether to apply decay in a discrete staircase, as opposed to continuous, fashion.
- `name`: String. Optional name of the operation. Defaults to 'ExponentialTimeDecay'.

Returns:

A scalar `Tensor` of the same type as `learning_rate`. The decayed learning rate.

Raises:

- `ValueError`: if `global_step` is not supplied.
