tf.feature_column.numeric_column

```
numeric_column(
    key,
    shape=(1,),
    default_value=None,
    dtype=tf.float32,
    normalizer_fn=None
)
```

Defined in `tensorflow/python/feature_column/feature_column.py`.

Represents real valued or numerical features.

Example:

```
price = numeric_column('price')
columns = [price, ...]
features = tf.parse_example(..., features=make_parse_example_spec(columns))
dense_tensor = input_layer(features, columns)

# or
bucketized_price = bucketized_column(price, boundaries=[...])
columns = [bucketized_price, ...]
features = tf.parse_example(..., features=make_parse_example_spec(columns))
linear_prediction = linear_model(features, columns)
```

Args:

- `key`: A unique string identifying the input feature. It is used as the column name and the dictionary key for feature parsing configs, feature `Tensor` objects, and feature columns.
- `shape`: An iterable of integers specifies the shape of the `Tensor`. An integer can be given which means a single dimension `Tensor` with given width. The `Tensor` representing the column will have the shape of [batch_size] + `shape`.
- `default_value`: A single value compatible with `dtype` or an iterable of values compatible with `dtype` which the column takes on during `tf.Example` parsing if data is missing. A default value of `None` will cause `tf.parse_example` to fail if an example does not contain this column. If a single value is provided, the same value will be applied as the default value for every item. If an iterable of values is provided, the shape of the `default_value` should be equal to the given `shape`.
- `dtype`: defines the type of values. Default value is `tf.float32`. Must be a non-quantized, real integer or floating point type.
- `normalizer_fn`: If not `None`, a function that can be used to normalize the value of the tensor after `default_value` is applied for parsing. Normalizer function takes the input `Tensor` as its argument, and returns the output `Tensor`. (e.g. lambda x: (x - 3.0) / 4.2). Please note that even though the most common use case of this function is normalization, it can be used for any kind of Tensorflow transformations.

Returns:

A `_NumericColumn`.

Raises:

- `TypeError`: if any dimension in shape is not an int
- `ValueError`: if any dimension in shape is not a positive integer
- `TypeError`: if **`default_value`** is an iterable but not compatible with **`shape`**
- `TypeError`: if **`default_value`** is not compatible with **`dtype`**.
- `ValueError`: if **`dtype`** is not convertible to **`tf.float32`**.

Last updated November 2, 2017.