C2_W4_Lab_3_Images

August 20, 2022

1 Ungraded Lab: Feature Engineering with Images

In this optional notebook, you will be looking at how to prepare features with an image dataset, particularly CIFAR-10. You will mostly go through the same steps but you will need to add parser functions in your transform module to successfully read and convert the data. As with the previous notebooks, we will just go briefly over the early stages of the pipeline so you can focus on the Transform component.

Let's begin!

1.1 Imports

[1]:
```
import os
import pprint
import tempfile
import urllib

import absl
import tensorflow as tf
tf.get_logger().propagate = False
pp = pprint.PrettyPrinter()

from tfx import v1 as tfx

from tfx.orchestration.experimental.interactive.interactive_context import␣
↪InteractiveContext
from tfx.types import Channel

from google.protobuf.json_format import MessageToDict

print('TensorFlow version: {}'.format(tf.__version__))
print('TFX version: {}'.format(tfx.__version__))
```

```
TensorFlow version: 2.6.0
TFX version: 1.3.0
```

1.2 Set up pipeline paths

[2]:
```
# Location of the pipeline metadata store
_pipeline_root = './pipeline/'

# Data files directory
_data_root = './data/cifar10'

# Path to the training data
_data_filepath = os.path.join(_data_root, 'train.tfrecord')
```

1.3 Download example data

We will download the training split of the CIFAR-10 dataset and save it to the `_data_filepath`. Take note that this is already in TFRecord format so we won't need to convert it when we use `ExampleGen` later.

[3]:
```
# Create data folder for the images
!mkdir -p {_data_root}

# URL of the hosted dataset
DATA_PATH = 'https://raw.githubusercontent.com/tensorflow/tfx/v0.21.4/tfx/
↪examples/cifar10/data/train.tfrecord'

# Download the dataset and save locally
urllib.request.urlretrieve(DATA_PATH, _data_filepath)
```

[3]:
```
('./data/cifar10/train.tfrecord', <http.client.HTTPMessage at 0x7f65803489a0>)
```

1.4 Create the InteractiveContext

[4]:
```
# Initialize the InteractiveContext
context = InteractiveContext(pipeline_root=_pipeline_root)
```

```
WARNING:absl:InteractiveContext metadata_connection_config not provided: using
SQLite ML Metadata database at ./pipeline/metadata.sqlite.
```

1.5 Run TFX components interactively

1.5.1 ExampleGen

As mentioned earlier, the dataset is already in TFRecord format so, unlike the previous TFX labs, there is no need to convert it when we ingest the data. You can simply import it with ImportExampleGen and here is the syntax and modules for that.

[5]:
```
# Ingest the data through ExampleGen
example_gen = tfx.components.ImportExampleGen(input_base=_data_root)
```

```python
# Run the component
context.run(example_gen)
```

```
WARNING:root:Make sure that locally built Python SDK docker image has Python 3.8
interpreter.
```

[5]:
```
ExecutionResult(
    component_id: ImportExampleGen
    execution_id: 1
    outputs:
        examples: Channel(
            type_name: Examples
            artifacts: [Artifact(artifact: id: 1
        type_id: 14
        uri: "./pipeline/ImportExampleGen/examples/1"
        properties {
          key: "split_names"
          value {
            string_value: "[\"train\", \"eval\"]"
          }
        }
        custom_properties {
          key: "file_format"
          value {
            string_value: "tfrecords_gzip"
          }
        }
        custom_properties {
          key: "input_fingerprint"
          value {
            string_value: "split:single_split,num_files:1,total_bytes:67847507,x
or_checksum:1661021317,sum_checksum:1661021317"
          }
        }
        custom_properties {
          key: "payload_format"
          value {
            string_value: "FORMAT_TF_EXAMPLE"
          }
        }
        custom_properties {
          key: "span"
          value {
            int_value: 0
          }
        }
        custom_properties {
```

```
    key: "state"
    value {
      string_value: "published"
    }
  }
  custom_properties {
    key: "tfx_version"
    value {
      string_value: "1.3.0"
    }
  }
  state: LIVE
  , artifact_type: id: 14
  name: "Examples"
  properties {
    key: "span"
    value: INT
  }
  properties {
    key: "split_names"
    value: STRING
  }
  properties {
    key: "version"
    value: INT
  }
  )]
      additional_properties: {}
      additional_custom_properties: {}
  ))
```

As usual, this component produces two artifacts, training examples and evaluation examples:

[6]:
```python
# Print split names and URI
artifact = example_gen.outputs['examples'].get()[0]
print(artifact.split_names, artifact.uri)
```

```
["train", "eval"] ./pipeline/ImportExampleGen/examples/1
```

You can also take a look at the first three training examples ingested by using the `tf.io.parse_single_example()` method from the tf.io module. See how it is setup in the cell below.

[7]:
```python
import IPython.display as display

# Get the URI of the output artifact representing the training examples, which
 ↪is a directory
```

```python
train_uri = os.path.join(example_gen.outputs['examples'].get()[0].uri, 'Split-train')

# Get the list of files in this directory (all compressed TFRecord files)
tfrecord_filenames = [os.path.join(train_uri, name)
                      for name in os.listdir(train_uri)]

# Create a `TFRecordDataset` to read these files
dataset = tf.data.TFRecordDataset(tfrecord_filenames, compression_type="GZIP")

# Description per example
image_feature_description = {
    'label': tf.io.FixedLenFeature([], tf.int64),
    'image_raw': tf.io.FixedLenFeature([], tf.string),
}

# Image parser function
def _parse_image_function(example_proto):
  # Parse the input tf.Example proto using the dictionary above.
  return tf.io.parse_single_example(example_proto, image_feature_description)

# Map the parser to the dataset
parsed_image_dataset = dataset.map(_parse_image_function)

# Display the first three images
for features in parsed_image_dataset.take(3):
    image_raw = features['image_raw'].numpy()
    display.display(display.Image(data=image_raw))
    pprint.pprint('Class ID: {}'.format(features['label'].numpy()))
```

```
'Class ID: 1'
```

```
'Class ID: 8'
```

```
'Class ID: 3'
```

1.5.2 StatisticsGen

Next, you will generate the statistics so you can infer a schema in the next step. You can also look at the visualization of the statistics. As you might expect with CIFAR-10, there is a column for the image and another column for the numeric label.

[8]:
```
# Run StatisticsGen
statistics_gen = tfx.components.StatisticsGen(
    examples=example_gen.outputs['examples'])

context.run(statistics_gen)
```

```
WARNING:root:Make sure that locally built Python SDK docker image has Python 3.8
interpreter.
```

[8]:
```
ExecutionResult(
    component_id: StatisticsGen
    execution_id: 2
    outputs:
        statistics: Channel(
            type_name: ExampleStatistics
            artifacts: [Artifact(artifact: id: 2
        type_id: 16
        uri: "./pipeline/StatisticsGen/statistics/2"
        properties {
          key: "split_names"
          value {
            string_value: "[\"train\", \"eval\"]"
          }
        }
        custom_properties {
          key: "name"
          value {
            string_value: "statistics"
          }
        }
        custom_properties {
          key: "producer_component"
          value {
            string_value: "StatisticsGen"
          }
        }
        custom_properties {
          key: "state"
          value {
            string_value: "published"
          }
        }
```

```
    custom_properties {
      key: "tfx_version"
      value {
        string_value: "1.3.0"
      }
    }
    state: LIVE
    , artifact_type: id: 16
    name: "ExampleStatistics"
    properties {
      key: "span"
      value: INT
    }
    properties {
      key: "split_names"
      value: STRING
    }
    )]
        additional_properties: {}
        additional_custom_properties: {}
    ))
```

[9]:
```python
# Visualize the results
context.show(statistics_gen.outputs['statistics'])
```

```
<IPython.core.display.HTML object>

<IPython.core.display.HTML object>

<IPython.core.display.HTML object>

<IPython.core.display.HTML object>

<IPython.core.display.HTML object>
```

1.5.3 SchemaGen

Here, you pass in the statistics to generate the Schema. For the version of TFX you are using, you will have to explicitly set `infer_feature_shape=True` so the downstream TFX components (e.g. Transform) will parse input as a `Tensor` and not `SparseTensor`. If not set, you will have compatibility issues later when you run the transform.

[10]:
```python
# Run SchemaGen
schema_gen = tfx.components.SchemaGen(
      statistics=statistics_gen.outputs['statistics'], infer_feature_shape=True)
context.run(schema_gen)
```

[10]:
```
ExecutionResult(
    component_id: SchemaGen
    execution_id: 3
```

```
    outputs:
        schema: Channel(
            type_name: Schema
            artifacts: [Artifact(artifact: id: 3
        type_id: 18
        uri: "./pipeline/SchemaGen/schema/3"
        custom_properties {
          key: "name"
          value {
            string_value: "schema"
          }
        }
        custom_properties {
          key: "producer_component"
          value {
            string_value: "SchemaGen"
          }
        }
        custom_properties {
          key: "state"
          value {
            string_value: "published"
          }
        }
        custom_properties {
          key: "tfx_version"
          value {
            string_value: "1.3.0"
          }
        }
        state: LIVE
        , artifact_type: id: 18
        name: "Schema"
        )]
            additional_properties: {}
            additional_custom_properties: {}
        ))
```

[11]:
```
# Visualize the results
context.show(schema_gen.outputs['schema'])
```

```
<IPython.core.display.HTML object>

               Type  Presence Valency Domain
Feature name
'image_raw'   BYTES  required               -
'label'         INT  required               -
```

1.5.4 ExampleValidator

`ExampleValidator` is not required but you can still run it just to make sure that there are no anomalies.

[12]:
```python
# Run ExampleValidator
example_validator = tfx.components.ExampleValidator(
    statistics=statistics_gen.outputs['statistics'],
    schema=schema_gen.outputs['schema'])
context.run(example_validator)
```

[12]:
```
ExecutionResult(
    component_id: ExampleValidator
    execution_id: 4
    outputs:
        anomalies: Channel(
            type_name: ExampleAnomalies
            artifacts: [Artifact(artifact: id: 4
        type_id: 20
        uri: "./pipeline/ExampleValidator/anomalies/4"
        properties {
          key: "split_names"
          value {
            string_value: "[\"train\", \"eval\"]"
          }
        }
        custom_properties {
          key: "name"
          value {
            string_value: "anomalies"
          }
        }
        custom_properties {
          key: "producer_component"
          value {
            string_value: "ExampleValidator"
          }
        }
        custom_properties {
          key: "state"
          value {
            string_value: "published"
          }
        }
        custom_properties {
          key: "tfx_version"
          value {
            string_value: "1.3.0"
```

```
    }
  }
  state: LIVE
  , artifact_type: id: 20
  name: "ExampleAnomalies"
  properties {
    key: "span"
    value: INT
  }
  properties {
    key: "split_names"
    value: STRING
  }
  )]
      additional_properties: {}
      additional_custom_properties: {}
  ))
```

[13]:

```
# Visualize the results. There should be no anomalies.
context.show(example_validator.outputs['anomalies'])
```

```
<IPython.core.display.HTML object>

<IPython.core.display.HTML object>

<IPython.core.display.HTML object>

<IPython.core.display.HTML object>

<IPython.core.display.HTML object>
```

1.5.5 Transform

To successfully transform the raw image, you need to parse the current bytes format and convert it to a tensor. For that, you can use the tf.image.decode_image() function. The transform module below utilizes this and converts the image to a `(32,32,3)` shaped float tensor. It also scales the pixels and converts the labels to one-hot tensors. The output features should then be ready to pass on to a model that accepts this format.

[14]:

```
_transform_module_file = 'cifar10_transform.py'
```

[15]:

```
%%writefile {_transform_module_file}

import tensorflow as tf
import tensorflow_transform as tft

# Keys
_LABEL_KEY = 'label'
_IMAGE_KEY = 'image_raw'
```

```python
def _transformed_name(key):
    return key + '_xf'

def _image_parser(image_str):
    '''converts the images to a float tensor'''
    image = tf.image.decode_image(image_str, channels=3)
    image = tf.reshape(image, (32, 32, 3))
    image = tf.cast(image, tf.float32)
    return image


def _label_parser(label_id):
    '''one hot encodes the labels'''
    label = tf.one_hot(label_id, 10)
    return label


def preprocessing_fn(inputs):
    """tf.transform's callback function for preprocessing inputs.
    Args:
        inputs: map from feature keys to raw not-yet-transformed features.
    Returns:
        Map from string feature key to transformed feature operations.
    """

    # Convert the raw image and labels to a float array and
    # one-hot encoded labels, respectively.
    with tf.device("/cpu:0"):
        outputs = {
            _transformed_name(_IMAGE_KEY):
                tf.map_fn(
                    _image_parser,
                    tf.squeeze(inputs[_IMAGE_KEY], axis=1),
                    dtype=tf.float32),
            _transformed_name(_LABEL_KEY):
                tf.map_fn(
                    _label_parser,
                    tf.squeeze(inputs[_LABEL_KEY], axis=1),
                    dtype=tf.float32)
        }

    # scale the pixels from 0 to 1
    outputs[_transformed_name(_IMAGE_KEY)] = tft.
↪scale_to_0_1(outputs[_transformed_name(_IMAGE_KEY)])

    return outputs
```

```
Writing cifar10_transform.py
```

Now, we pass in this feature engineering code to the `Transform` component and run it to transform your data.

[16]:
```python
# Ignore TF warning messages
tf.get_logger().setLevel('ERROR')

# Setup the Transform component
transform = tfx.components.Transform(
    examples=example_gen.outputs['examples'],
    schema=schema_gen.outputs['schema'],
    module_file=os.path.abspath(_transform_module_file))

# Run the component
context.run(transform)
```

```
WARNING:root:This output type hint will be ignored and not used for type-
checking purposes. Typically, output type hints for a PTransform are single (or
nested) types wrapped by a PCollection, PDone, or None. Got: Tuple[Dict[str,
Union[NoneType, _Dataset]], Union[Dict[str, Dict[str, PCollection]], NoneType],
int] instead.
WARNING:root:This output type hint will be ignored and not used for type-
checking purposes. Typically, output type hints for a PTransform are single (or
nested) types wrapped by a PCollection, PDone, or None. Got: Tuple[Dict[str,
Union[NoneType, _Dataset]], Union[Dict[str, Dict[str, PCollection]], NoneType],
int] instead.
WARNING:root:Make sure that locally built Python SDK docker image has Python 3.8
interpreter.
```

[16]:
```
ExecutionResult(
    component_id: Transform
    execution_id: 5
    outputs:
        transform_graph: Channel(
            type_name: TransformGraph
            artifacts: [Artifact(artifact: id: 5
        type_id: 22
        uri: "./pipeline/Transform/transform_graph/5"
        custom_properties {
          key: "name"
          value {
            string_value: "transform_graph"
          }
        }
        custom_properties {
          key: "producer_component"
          value {
```

```
    string_value: "Transform"
  }
}
custom_properties {
  key: "state"
  value {
    string_value: "published"
  }
}
custom_properties {
  key: "tfx_version"
  value {
    string_value: "1.3.0"
  }
}
state: LIVE
, artifact_type: id: 22
name: "TransformGraph"
)]
    additional_properties: {}
    additional_custom_properties: {}
)
transformed_examples: Channel(
    type_name: Examples
    artifacts: [Artifact(artifact: id: 6
type_id: 14
uri: "./pipeline/Transform/transformed_examples/5"
properties {
  key: "split_names"
  value {
    string_value: "[\"train\", \"eval\"]"
  }
}
custom_properties {
  key: "name"
  value {
    string_value: "transformed_examples"
  }
}
custom_properties {
  key: "producer_component"
  value {
    string_value: "Transform"
  }
}
custom_properties {
  key: "state"
```

```
  value {
    string_value: "published"
  }
}
custom_properties {
  key: "tfx_version"
  value {
    string_value: "1.3.0"
  }
}
state: LIVE
, artifact_type: id: 14
name: "Examples"
properties {
  key: "span"
  value: INT
}
properties {
  key: "split_names"
  value: STRING
}
properties {
  key: "version"
  value: INT
}
)]
    additional_properties: {}
    additional_custom_properties: {}
)
updated_analyzer_cache: Channel(
    type_name: TransformCache
    artifacts: [Artifact(artifact: id: 7
type_id: 23
uri: "./pipeline/Transform/updated_analyzer_cache/5"
custom_properties {
  key: "name"
  value {
    string_value: "updated_analyzer_cache"
  }
}
custom_properties {
  key: "producer_component"
  value {
    string_value: "Transform"
  }
}
custom_properties {
```

```
  key: "state"
  value {
    string_value: "published"
  }
}
custom_properties {
  key: "tfx_version"
  value {
    string_value: "1.3.0"
  }
}
state: LIVE
, artifact_type: id: 23
name: "TransformCache"
)]
    additional_properties: {}
    additional_custom_properties: {}
)
pre_transform_schema: Channel(
    type_name: Schema
    artifacts: [Artifact(artifact: id: 8
type_id: 18
uri: "./pipeline/Transform/pre_transform_schema/5"
custom_properties {
  key: "name"
  value {
    string_value: "pre_transform_schema"
  }
}
custom_properties {
  key: "producer_component"
  value {
    string_value: "Transform"
  }
}
custom_properties {
  key: "state"
  value {
    string_value: "published"
  }
}
custom_properties {
  key: "tfx_version"
  value {
    string_value: "1.3.0"
  }
}
```

```
state: LIVE
, artifact_type: id: 18
name: "Schema"
)]
    additional_properties: {}
    additional_custom_properties: {}
)
pre_transform_stats: Channel(
    type_name: ExampleStatistics
    artifacts: [Artifact(artifact: id: 9
type_id: 16
uri: "./pipeline/Transform/pre_transform_stats/5"
custom_properties {
  key: "name"
  value {
    string_value: "pre_transform_stats"
  }
}
custom_properties {
  key: "producer_component"
  value {
    string_value: "Transform"
  }
}
custom_properties {
  key: "state"
  value {
    string_value: "published"
  }
}
custom_properties {
  key: "tfx_version"
  value {
    string_value: "1.3.0"
  }
}
state: LIVE
, artifact_type: id: 16
name: "ExampleStatistics"
properties {
  key: "span"
  value: INT
}
properties {
  key: "split_names"
  value: STRING
}
```

```
)]
    additional_properties: {}
    additional_custom_properties: {}
)
post_transform_schema: Channel(
    type_name: Schema
    artifacts: [Artifact(artifact: id: 10
type_id: 18
uri: "./pipeline/Transform/post_transform_schema/5"
custom_properties {
  key: "name"
  value {
    string_value: "post_transform_schema"
  }
}
custom_properties {
  key: "producer_component"
  value {
    string_value: "Transform"
  }
}
custom_properties {
  key: "state"
  value {
    string_value: "published"
  }
}
custom_properties {
  key: "tfx_version"
  value {
    string_value: "1.3.0"
  }
}
state: LIVE
, artifact_type: id: 18
name: "Schema"
)]
    additional_properties: {}
    additional_custom_properties: {}
)
post_transform_stats: Channel(
    type_name: ExampleStatistics
    artifacts: [Artifact(artifact: id: 11
type_id: 16
uri: "./pipeline/Transform/post_transform_stats/5"
custom_properties {
  key: "name"
```

```
  value {
    string_value: "post_transform_stats"
  }
}
custom_properties {
  key: "producer_component"
  value {
    string_value: "Transform"
  }
}
custom_properties {
  key: "state"
  value {
    string_value: "published"
  }
}
custom_properties {
  key: "tfx_version"
  value {
    string_value: "1.3.0"
  }
}
state: LIVE
, artifact_type: id: 16
name: "ExampleStatistics"
properties {
  key: "span"
  value: INT
}
properties {
  key: "split_names"
  value: STRING
}
)]
    additional_properties: {}
    additional_custom_properties: {}
)
post_transform_anomalies: Channel(
    type_name: ExampleAnomalies
    artifacts: [Artifact(artifact: id: 12
type_id: 20
uri: "./pipeline/Transform/post_transform_anomalies/5"
custom_properties {
  key: "name"
  value {
    string_value: "post_transform_anomalies"
  }
```

```
}
custom_properties {
  key: "producer_component"
  value {
    string_value: "Transform"
  }
}
custom_properties {
  key: "state"
  value {
    string_value: "published"
  }
}
custom_properties {
  key: "tfx_version"
  value {
    string_value: "1.3.0"
  }
}
state: LIVE
, artifact_type: id: 20
name: "ExampleAnomalies"
properties {
  key: "span"
  value: INT
}
properties {
  key: "split_names"
  value: STRING
}
)]
    additional_properties: {}
    additional_custom_properties: {}
))
```

1.5.6 Preview the results

Now that the Transform component is finished, you can preview how the transformed images and labels look like. You can use the same sequence and helper function from previous labs.

[17]:
```
# Get the URI of the output artifact representing the transformed examples,␣
↪which is a directory
train_uri = os.path.join(transform.outputs['transformed_examples'].get()[0].
↪uri, 'Split-train')

# Get the list of files in this directory (all compressed TFRecord files)
tfrecord_filenames = [os.path.join(train_uri, name)
```

```
                        for name in os.listdir(train_uri)]

# Create a `TFRecordDataset` to read these files
dataset = tf.data.TFRecordDataset(tfrecord_filenames, compression_type="GZIP")
```

[18]:
```
# Define a helper function to get individual examples
def get_records(dataset, num_records):
    '''Extracts records from the given dataset.
    Args:
        dataset (TFRecordDataset): dataset saved by ExampleGen
        num_records (int): number of records to preview
    '''

    # initialize an empty list
    records = []

    # Use the `take()` method to specify how many records to get
    for tfrecord in dataset.take(num_records):

        # Get the numpy property of the tensor
        serialized_example = tfrecord.numpy()

        # Initialize a `tf.train.Example()` to read the serialized data
        example = tf.train.Example()

        # Read the example data (output is a protocol buffer message)
        example.ParseFromString(serialized_example)

        # convert the protocol bufffer message to a Python dictionary
        example_dict = (MessageToDict(example))

        # append to the records list
        records.append(example_dict)

    return records
```

You should see from the output of the cell below that the transformed raw image (i.e. `image_raw_xf`) now has a float array that is scaled from 0 to 1. Similarly, you'll see that the transformed label (i.e. `label_xf`) is now one-hot encoded.

[19]:
```
# Get 1 record from the dataset
sample_records = get_records(dataset, 1)

# Print the output
pp.pprint(sample_records)
```


```
[{'features': {'feature': {'image_raw_xf': {'floatList': {'value': [0.16470589,
                                                                    0.09019608,
```

```
0.0627451,
0.2627451,
0.2,
0.14901961,
0.28627452,
0.22352941,
0.16078432,
0.29803923,
0.2509804,
0.16470589,
0.36862746,
0.3019608,
0.23921569,
0.36862746,
0.29803923,
0.24313726,
0.28627452,
0.21176471,
0.18431373,
0.32941177,
0.2627451,
0.23137255,
0.3254902,
0.2784314,
0.22352941,
0.12941177,
0.09803922,
0.047058824,
0.07450981,
0.05490196,
0.043137256,
0.09411765,
0.078431375,
0.06666667,
0.15294118,
0.1254902,
0.101960786,
0.22745098,
0.1882353,
0.14901961,
0.16862746,
0.101960786,
0.07058824,
0.16862746,
0.09019608,
0.0627451,
0.23529412,
0.13725491,
```

```
0.14901961,
0.12941177,
0.050980393,
0.05490196,
0.14901961,
0.09803922,
0.07450981,
0.15294118,
0.12941177,
0.08235294,
0.25882354,
0.2509804,
0.2,
0.21960784,
0.22352941,
0.16078432,
0.24705882,
0.23137255,
0.18431373,
0.22745098,
0.20392157,
0.15686275,
0.23529412,
0.18039216,
0.14509805,
0.2509804,
0.19607843,
0.14509805,
0.24705882,
0.21568628,
0.13333334,
0.23529412,
0.19607843,
0.09019608,
0.3529412,
0.27058825,
0.16470589,
0.5019608,
0.40784314,
0.30588236,
0.40392157,
0.31764707,
0.23529412,
0.36078432,
0.2784314,
0.20392157,
0.22352941,
0.14509805,
```

```
0.10980392,
0.3254902,
0.25490198,
0.20784314,
0.2901961,
0.23921569,
0.1764706,
0.23921569,
0.19607843,
0.1254902,
0.27450982,
0.21176471,
0.15294118,
0.29803923,
0.23529412,
0.1764706,
0.36862746,
0.29803923,
0.2509804,
0.40784314,
0.34509805,
0.29411766,
0.34117648,
0.2901961,
0.22745098,
0.1254902,
0.08627451,
0.039215688,
0.09411765,
0.0627451,
0.050980393,
0.08627451,
0.07058824,
0.05882353,
0.14117648,
0.12941177,
0.09411765,
0.21568628,
0.19215687,
0.14509805,
0.22352941,
0.1764706,
0.12941177,
0.2,
0.13333334,
0.101960786,
0.21960784,
0.14509805,
```

```
0.12941177,
0.14901961,
0.09019608,
0.07058824,
0.101960786,
0.06666667,
0.03137255,
0.09019608,
0.07450981,
0.03137255,
0.13333334,
0.13333334,
0.08627451,
0.16470589,
0.16470589,
0.11764706,
0.14901961,
0.14117648,
0.09411765,
0.105882354,
0.09019608,
0.043137256,
0.08235294,
0.047058824,
0.011764706,
0.16470589,
0.1254902,
0.08627451,
0.23529412,
0.2,
0.13333334,
0.28235295,
0.23529412,
0.14901961,
0.31764707,
0.23137255,
0.13725491,
0.40784314,
0.3137255,
0.21176471,
0.49803922,
0.41960785,
0.3254902,
0.41568628,
0.34509805,
0.2509804,
0.3372549,
0.2509804,
```

```
0.19607843,
0.40784314,
0.3372549,
0.28235295,
0.3254902,
0.28627452,
0.23921569,
0.18039216,
0.14901961,
0.105882354,
0.20784314,
0.16078432,
0.11372549,
0.27058825,
0.21960784,
0.15686275,
0.42352942,
0.3647059,
0.28235295,
0.4627451,
0.39607844,
0.31764707,
0.30980393,
0.24313726,
0.17254902,
0.13333334,
0.06666667,
0.02745098,
0.07450981,
0.039215688,
0.019607844,
0.06666667,
0.05490196,
0.03529412,
0.12156863,
0.12941177,
0.07450981,
0.17254902,
0.1764706,
0.105882354,
0.19215687,
0.16862746,
0.11372549,
0.14509805,
0.11372549,
0.0627451,
0.16470589,
0.12941177,
```

```
0.07058824,
0.15294118,
0.12156863,
0.07058824,
0.10980392,
0.101960786,
0.050980393,
0.07450981,
0.08235294,
0.039215688,
0.08235294,
0.08627451,
0.0627451,
0.14117648,
0.13725491,
0.11764706,
0.10980392,
0.10980392,
0.07058824,
0.0627451,
0.0627451,
0.015686275,
0.043137256,
0.047058824,
0.015686275,
0.09803922,
0.08627451,
0.05882353,
0.2,
0.16078432,
0.1254902,
0.27058825,
0.2,
0.14509805,
0.32941177,
0.23137255,
0.15294118,
0.30980393,
0.21568628,
0.11372549,
0.34901962,
0.27450982,
0.15686275,
0.40392157,
0.34509805,
0.22352941,
0.30588236,
0.21960784,
```

```
0.16470589,
0.2784314,
0.21568628,
0.15686275,
0.23137255,
0.20784314,
0.16078432,
0.14901961,
0.13725491,
0.101960786,
0.22745098,
0.2,
0.16078432,
0.25490198,
0.21568628,
0.16862746,
0.24313726,
0.21176471,
0.12941177,
0.25490198,
0.21176471,
0.1254902,
0.23921569,
0.1764706,
0.11372549,
0.18039216,
0.1254902,
0.08235294,
0.11372549,
0.08627451,
0.0627451,
0.09803922,
0.09803922,
0.06666667,
0.09803922,
0.105882354,
0.050980393,
0.10980392,
0.12156863,
0.05490196,
0.105882354,
0.09803922,
0.047058824,
0.13725491,
0.11372549,
0.05882353,
0.1254902,
0.09803922,
```

```
0.03529412,
0.13725491,
0.1254902,
0.05882353,
0.14509805,
0.15686275,
0.09019608,
0.101960786,
0.11764706,
0.0627451,
0.08627451,
0.08627451,
0.078431375,
0.101960786,
0.09803922,
0.09019608,
0.10980392,
0.09803922,
0.078431375,
0.09019608,
0.078431375,
0.050980393,
0.06666667,
0.07058824,
0.047058824,
0.11372549,
0.101960786,
0.07450981,
0.25490198,
0.1882353,
0.16078432,
0.4117647,
0.31764707,
0.2784314,
0.48235294,
0.3764706,
0.3019608,
0.52156866,
0.42745098,
0.33333334,
0.5921569,
0.5254902,
0.41568628,
0.6901961,
0.6313726,
0.50980395,
0.23529412,
0.16078432,
```

```
0.105882354,
0.13725491,
0.09019608,
0.03529412,
0.11372549,
0.105882354,
0.05882353,
0.06666667,
0.078431375,
0.043137256,
0.12156863,
0.11764706,
0.101960786,
0.14117648,
0.13725491,
0.11764706,
0.09411765,
0.101960786,
0.050980393,
0.09019608,
0.09411765,
0.039215688,
0.15294118,
0.12941177,
0.07450981,
0.15686275,
0.1254902,
0.08235294,
0.105882354,
0.09019608,
0.05490196,
0.101960786,
0.101960786,
0.07058824,
0.08235294,
0.08627451,
0.05490196,
0.08235294,
0.09019608,
0.047058824,
0.09411765,
0.08235294,
0.047058824,
0.15294118,
0.1254902,
0.09411765,
0.13333334,
0.09803922,
```

```
0.0627451,
0.13725491,
0.11372549,
0.06666667,
0.15294118,
0.15686275,
0.09411765,
0.12156863,
0.12941177,
0.07450981,
0.101960786,
0.09803922,
0.08235294,
0.09411765,
0.07058824,
0.078431375,
0.1254902,
0.08627451,
0.08235294,
0.14117648,
0.101960786,
0.09411765,
0.11372549,
0.09411765,
0.07058824,
0.15294118,
0.11372549,
0.078431375,
0.3372549,
0.23137255,
0.1882353,
0.50980395,
0.38431373,
0.3254902,
0.56078434,
0.4509804,
0.36862746,
0.6117647,
0.5137255,
0.42745098,
0.5764706,
0.5058824,
0.42745098,
0.53333336,
0.4745098,
0.39215687,
0.32156864,
0.22745098,
```

```
0.1254902,
0.18431373,
0.11764706,
0.015686275,
0.101960786,
0.07450981,
0.0,
0.07450981,
0.05490196,
0.0,
0.050980393,
0.023529412,
0.0,
0.05882353,
0.03529412,
0.0,
0.07058824,
0.07450981,
0.019607844,
0.08235294,
0.09019608,
0.03529412,
0.11372549,
0.09803922,
0.05490196,
0.10980392,
0.08235294,
0.050980393,
0.09803922,
0.078431375,
0.05490196,
0.078431375,
0.06666667,
0.039215688,
0.09019608,
0.09019608,
0.05882353,
0.09411765,
0.09411765,
0.0627451,
0.12156863,
0.10980392,
0.08235294,
0.11764706,
0.09803922,
0.07450981,
0.13725491,
0.10980392,
```

```
0.078431375,
0.1254902,
0.10980392,
0.06666667,
0.1254902,
0.12941177,
0.05882353,
0.1254902,
0.12941177,
0.06666667,
0.11764706,
0.09803922,
0.07450981,
0.12156863,
0.09019608,
0.08235294,
0.1254902,
0.078431375,
0.078431375,
0.14117648,
0.09803922,
0.08235294,
0.105882354,
0.07058824,
0.03529412,
0.13725491,
0.09019608,
0.03529412,
0.28235295,
0.19215687,
0.13725491,
0.42352942,
0.31764707,
0.2509804,
0.5411765,
0.45490196,
0.3647059,
0.6156863,
0.5372549,
0.44313726,
0.5529412,
0.4745098,
0.38039216,
0.45490196,
0.3882353,
0.28627452,
0.41960785,
0.27058825,
```

```
0.08627451,
0.29411766,
0.16078432,
0.0,
0.22745098,
0.1254902,
0.0,
0.30980393,
0.21960784,
0.023529412,
0.34901962,
0.23921569,
0.05490196,
0.24313726,
0.15294118,
0.0,
0.18039216,
0.13725491,
0.02745098,
0.16862746,
0.14117648,
0.07058824,
0.1254902,
0.08627451,
0.050980393,
0.16078432,
0.11764706,
0.101960786,
0.18431373,
0.14509805,
0.13725491,
0.105882354,
0.078431375,
0.05490196,
0.11372549,
0.09411765,
0.07058824,
0.09411765,
0.08235294,
0.047058824,
0.1254902,
0.11372549,
0.078431375,
0.105882354,
0.105882354,
0.06666667,
0.13725491,
0.12941177,
```

```
0.08235294,
0.14901961,
0.14117648,
0.09019608,
0.15686275,
0.15294118,
0.07450981,
0.13333334,
0.12156863,
0.047058824,
0.11764706,
0.09411765,
0.039215688,
0.13725491,
0.101960786,
0.06666667,
0.12941177,
0.09019608,
0.05490196,
0.105882354,
0.06666667,
0.02745098,
0.19607843,
0.1764706,
0.101960786,
0.16078432,
0.13333334,
0.05882353,
0.23921569,
0.19607843,
0.1254902,
0.40784314,
0.35686275,
0.2901961,
0.33333334,
0.28627452,
0.2,
0.30980393,
0.2509804,
0.13725491,
0.4745098,
0.3882353,
0.23529412,
0.52156866,
0.42745098,
0.24705882,
0.48235294,
0.30588236,
```

```
0.08235294,
0.48235294,
0.31764707,
0.09019608,
0.44705883,
0.29803923,
0.05490196,
0.49411765,
0.34509805,
0.09803922,
0.5568628,
0.39607844,
0.15294118,
0.40784314,
0.26666668,
0.0627451,
0.22352941,
0.14509805,
0.015686275,
0.15686275,
0.11372549,
0.043137256,
0.101960786,
0.0627451,
0.023529412,
0.18039216,
0.14509805,
0.11764706,
0.1882353,
0.15294118,
0.1254902,
0.078431375,
0.050980393,
0.019607844,
0.105882354,
0.09019608,
0.05490196,
0.105882354,
0.09019608,
0.047058824,
0.12156863,
0.105882354,
0.0627451,
0.12941177,
0.11764706,
0.05882353,
0.17254902,
0.16470589,
```

```
0.08235294,
0.22745098,
0.21176471,
0.11372549,
0.2509804,
0.22745098,
0.13333334,
0.17254902,
0.14117648,
0.05882353,
0.15294118,
0.11764706,
0.050980393,
0.1882353,
0.15294118,
0.09411765,
0.2,
0.16078432,
0.11372549,
0.15686275,
0.12156863,
0.0627451,
0.33333334,
0.30980393,
0.21568628,
0.18039216,
0.16470589,
0.06666667,
0.12156863,
0.09411765,
0.023529412,
0.15686275,
0.1254902,
0.050980393,
0.11372549,
0.06666667,
0.0,
0.3764706,
0.30588236,
0.16470589,
0.57254905,
0.46666667,
0.25882354,
0.5568628,
0.43137255,
0.19215687,
0.4,
0.21568628,
```

```
0.011764706,
0.4392157,
0.2627451,
0.047058824,
0.4627451,
0.28627452,
0.05490196,
0.4627451,
0.2901961,
0.043137256,
0.45490196,
0.26666668,
0.023529412,
0.35686275,
0.19215687,
0.0,
0.16862746,
0.06666667,
0.0,
0.105882354,
0.05490196,
0.0,
0.1254902,
0.09803922,
0.03529412,
0.23921569,
0.22745098,
0.16078432,
0.12156863,
0.1254902,
0.0627451,
0.0627451,
0.06666667,
0.003921569,
0.09411765,
0.08627451,
0.039215688,
0.101960786,
0.078431375,
0.03137255,
0.12156863,
0.09411765,
0.03137255,
0.18431373,
0.14509805,
0.047058824,
0.34117648,
0.29411766,
```

```
0.14509805,
0.49019608,
0.43137255,
0.27058825,
0.46666667,
0.39215687,
0.2627451,
0.2509804,
0.18431373,
0.08235294,
0.18431373,
0.1254902,
0.050980393,
0.22352941,
0.1764706,
0.11372549,
0.18039216,
0.14901961,
0.09803922,
0.13333334,
0.105882354,
0.03529412,
0.19607843,
0.16862746,
0.06666667,
0.25490198,
0.22745098,
0.11764706,
0.15686275,
0.12941177,
0.05882353,
0.15686275,
0.11372549,
0.03529412,
0.20784314,
0.1254902,
0.003921569,
0.40784314,
0.28627452,
0.101960786,
0.4862745,
0.33333334,
0.078431375,
0.47843137,
0.30588236,
0.019607844,
0.35686275,
0.18431373,
```

```
0.0,
0.39215687,
0.22352941,
0.023529412,
0.41960785,
0.24313726,
0.019607844,
0.4745098,
0.28627452,
0.043137256,
0.4745098,
0.27450982,
0.015686275,
0.52156866,
0.34117648,
0.09803922,
0.35686275,
0.21960784,
0.05490196,
0.15294118,
0.05882353,
0.0,
0.14901961,
0.09411765,
0.0,
0.15294118,
0.12156863,
0.0,
0.11764706,
0.105882354,
0.0,
0.12941177,
0.1254902,
0.007843138,
0.15294118,
0.13725491,
0.039215688,
0.16078432,
0.12941177,
0.047058824,
0.2,
0.15294118,
0.05882353,
0.24313726,
0.18039216,
0.050980393,
0.5058824,
0.41960785,
```

```
0.23529412,
0.8666667,
0.7647059,
0.57254905,
0.6039216,
0.49411765,
0.3372549,
0.3254902,
0.22745098,
0.105882354,
0.2509804,
0.17254902,
0.07450981,
0.24705882,
0.1882353,
0.105882354,
0.21568628,
0.18431373,
0.10980392,
0.17254902,
0.13333334,
0.03529412,
0.2,
0.14509805,
0.007843138,
0.3137255,
0.2509804,
0.10980392,
0.38039216,
0.32156864,
0.2,
0.23921569,
0.16470589,
0.03529412,
0.32156864,
0.19607843,
0.003921569,
0.4745098,
0.31764707,
0.078431375,
0.49411765,
0.30980393,
0.03529412,
0.52156866,
0.32941177,
0.03137255,
0.34901962,
0.20392157,
```

```
0.03137255,
0.3764706,
0.22745098,
0.043137256,
0.4,
0.23529412,
0.039215688,
0.43529412,
0.25490198,
0.011764706,
0.5176471,
0.31764707,
0.007843138,
0.62352943,
0.41960785,
0.09019608,
0.5882353,
0.39607844,
0.09411765,
0.4862745,
0.3137255,
0.02745098,
0.36078432,
0.22352941,
0.0,
0.26666668,
0.17254902,
0.0,
0.19215687,
0.12941177,
0.0,
0.14901961,
0.10980392,
0.0,
0.16862746,
0.1254902,
0.0,
0.17254902,
0.11764706,
0.0,
0.21176471,
0.13333334,
0.003921569,
0.30588236,
0.20392157,
0.05882353,
0.54901963,
0.42352942,
```

```
0.23137255,
0.80784315,
0.6666667,
0.4627451,
0.54901963,
0.42352942,
0.23921569,
0.3372549,
0.22352941,
0.06666667,
0.26666668,
0.1764706,
0.043137256,
0.23529412,
0.16078432,
0.03529412,
0.18039216,
0.12156863,
0.0,
0.1764706,
0.11764706,
0.0,
0.24705882,
0.15686275,
0.0,
0.3137255,
0.21176471,
0.0,
0.46666667,
0.3529412,
0.13333334,
0.44313726,
0.30980393,
0.0627451,
0.52156866,
0.34901962,
0.05490196,
0.57254905,
0.3882353,
0.08235294,
0.4862745,
0.29803923,
0.015686275,
0.4509804,
0.26666668,
0.0,
0.3019608,
0.19607843,
```

0.02745098,
0.34509805,
0.22352941,
0.050980393,
0.35686275,
0.21568628,
0.02745098,
0.4,
0.24313726,
0.011764706,
0.52156866,
0.34901962,
0.05490196,
0.64705884,
0.4509804,
0.101960786,
0.7019608,
0.48235294,
0.101960786,
0.73333335,
0.5176471,
0.12156863,
0.70980394,
0.52156866,
0.13725491,
0.6509804,
0.49803922,
0.14117648,
0.54901963,
0.42745098,
0.101960786,
0.44705883,
0.34509805,
0.043137256,
0.46666667,
0.3764706,
0.07058824,
0.4745098,
0.37254903,
0.08235294,
0.49803922,
0.3764706,
0.12156863,
0.6117647,
0.4745098,
0.22352941,
0.72156864,
0.5686275,

```
0.29803923,
0.7411765,
0.58431375,
0.30980393,
0.6431373,
0.49411765,
0.21568628,
0.5176471,
0.38039216,
0.11372549,
0.47058824,
0.34901962,
0.09803922,
0.4392157,
0.32941177,
0.08235294,
0.38431373,
0.28627452,
0.03529412,
0.4117647,
0.30588236,
0.03529412,
0.53333336,
0.39607844,
0.078431375,
0.5764706,
0.42745098,
0.09411765,
0.64705884,
0.49019608,
0.14901961,
0.65882355,
0.4862745,
0.14509805,
0.6627451,
0.4627451,
0.105882354,
0.654902,
0.4509804,
0.12156863,
0.5294118,
0.34117648,
0.05882353,
0.42745098,
0.2509804,
0.0,
0.2627451,
0.20784314,
```

```
0.03137255,
0.30588236,
0.23529412,
0.05490196,
0.32156864,
0.21568628,
0.03137255,
0.3647059,
0.24705882,
0.043137256,
0.38431373,
0.25882354,
0.03529412,
0.4862745,
0.32941177,
0.047058824,
0.6862745,
0.47058824,
0.078431375,
0.78039217,
0.5411765,
0.08235294,
0.77254903,
0.5568628,
0.0627451,
0.7921569,
0.5921569,
0.09411765,
0.8156863,
0.63529414,
0.15686275,
0.7882353,
0.62352943,
0.14901961,
0.8039216,
0.64705884,
0.16078432,
0.81960785,
0.65882355,
0.17254902,
0.84705883,
0.6627451,
0.2,
0.85490197,
0.6784314,
0.22745098,
0.8862745,
0.72156864,
```

```
0.2901961,
0.94509804,
0.78431374,
0.36078432,
0.9137255,
0.7490196,
0.30980393,
0.8352941,
0.6666667,
0.22352941,
0.8352941,
0.6666667,
0.22352941,
0.7921569,
0.6313726,
0.18431373,
0.7607843,
0.6,
0.15294118,
0.74509805,
0.5803922,
0.11764706,
0.7882353,
0.59607846,
0.12156863,
0.8039216,
0.60784316,
0.13333334,
0.79607844,
0.59607846,
0.13333334,
0.7254902,
0.52156866,
0.09411765,
0.69803923,
0.48235294,
0.11764706,
0.5647059,
0.35686275,
0.043137256,
0.4862745,
0.29411766,
0.03137255,
0.43137255,
0.2509804,
0.015686275,
0.24705882,
0.21960784,
```

```
0.047058824,
0.24705882,
0.20392157,
0.023529412,
0.27058825,
0.2,
0.011764706,
0.36078432,
0.27450982,
0.09019608,
0.36862746,
0.28235295,
0.12156863,
0.4,
0.27450982,
0.050980393,
0.6784314,
0.4745098,
0.08627451,
0.7921569,
0.5529412,
0.0627451,
0.81960785,
0.5882353,
0.039215688,
0.83137256,
0.6039216,
0.039215688,
0.84705883,
0.6313726,
0.07450981,
0.84313726,
0.63529414,
0.078431375,
0.84313726,
0.64705884,
0.07058824,
0.87058824,
0.6745098,
0.09803922,
0.91764706,
0.7176471,
0.14901961,
0.9098039,
0.7176471,
0.17254902,
0.9529412,
0.78431374,
```

```
0.27058825,
1.0,
0.8980392,
0.38431373,
0.98039216,
0.8,
0.2627451,
0.8980392,
0.7137255,
0.16470589,
0.8862745,
0.69411767,
0.14117648,
0.84705883,
0.654902,
0.101960786,
0.84313726,
0.6431373,
0.09019608,
0.827451,
0.61960787,
0.07058824,
0.84313726,
0.61960787,
0.05882353,
0.8235294,
0.6,
0.047058824,
0.7647059,
0.5529412,
0.023529412,
0.7490196,
0.54509807,
0.08235294,
0.64705884,
0.4509804,
0.10980392,
0.42352942,
0.23921569,
0.0,
0.41960785,
0.24313726,
0.011764706,
0.41960785,
0.2509804,
0.03529412,
0.24705882,
0.24313726,
```

```
0.07058824,
0.23137255,
0.21568628,
0.03137255,
0.2627451,
0.20784314,
0.015686275,
0.3137255,
0.24705882,
0.078431375,
0.3137255,
0.25882354,
0.15294118,
0.40784314,
0.3137255,
0.14117648,
0.6392157,
0.45490196,
0.09411765,
0.80784315,
0.5803922,
0.09019608,
0.81960785,
0.5882353,
0.047058824,
0.84705883,
0.60784316,
0.047058824,
0.85490197,
0.6156863,
0.05882353,
0.87058824,
0.6431373,
0.08235294,
0.8862745,
0.67058825,
0.101960786,
0.8980392,
0.7019608,
0.1254902,
0.90588236,
0.72156864,
0.16470589,
0.9372549,
0.76862746,
0.22352941,
0.9411765,
0.77254903,
```

```
0.23137255,
0.9843137,
0.8156863,
0.27058825,
0.9372549,
0.7529412,
0.19607843,
0.9137255,
0.7254902,
0.15294118,
0.89411765,
0.70980394,
0.12941177,
0.8666667,
0.67058825,
0.09411765,
0.8509804,
0.6431373,
0.08627451,
0.8509804,
0.627451,
0.07450981,
0.8392157,
0.6117647,
0.050980393,
0.8352941,
0.6117647,
0.050980393,
0.7764706,
0.5686275,
0.019607844,
0.7490196,
0.5647059,
0.10980392,
0.5529412,
0.40784314,
0.13333334,
0.39215687,
0.26666668,
0.08235294,
0.39607844,
0.2627451,
0.08235294,
0.38431373,
0.2509804,
0.0627451,
0.20784314,
0.21568628,
```

```
0.0627451,
0.23921569,
0.23529412,
0.0627451,
0.26666668,
0.22745098,
0.03529412,
0.3254902,
0.27058825,
0.11764706,
0.3254902,
0.2901961,
0.22352941,
0.6745098,
0.6,
0.47058824,
0.7176471,
0.5568628,
0.21176471,
0.8,
0.5882353,
0.10980392,
0.8235294,
0.6,
0.05490196,
0.8509804,
0.6156863,
0.043137256,
0.8627451,
0.62352943,
0.0627451,
0.87058824,
0.6431373,
0.078431375,
0.87058824,
0.65882355,
0.078431375,
0.8784314,
0.68235296,
0.09803922,
0.8666667,
0.6862745,
0.11764706,
0.92156863,
0.74509805,
0.18431373,
0.92156863,
0.7372549,
```

```
0.18039216,
0.91764706,
0.7372549,
0.16862746,
0.8901961,
0.7058824,
0.1254902,
0.8901961,
0.69411767,
0.10980392,
0.8862745,
0.6901961,
0.09803922,
0.8627451,
0.6666667,
0.08235294,
0.8666667,
0.6509804,
0.08235294,
0.84313726,
0.61960787,
0.05882353,
0.84705883,
0.61960787,
0.05490196,
0.827451,
0.6039216,
0.043137256,
0.8117647,
0.59607846,
0.039215688,
0.72156864,
0.5529412,
0.10980392,
0.5803922,
0.46666667,
0.24705882,
0.5176471,
0.42745098,
0.3019608,
0.39607844,
0.28627452,
0.12941177,
0.3882353,
0.26666668,
0.09411765,
0.15294118,
0.18039216,
```

```
0.05882353,
0.16470589,
0.16862746,
0.03529412,
0.2509804,
0.21568628,
0.047058824,
0.29803923,
0.24705882,
0.10980392,
0.32156864,
0.2901961,
0.23921569,
0.47843137,
0.41960785,
0.29803923,
0.5764706,
0.43529412,
0.09803922,
0.80784315,
0.6156863,
0.14117648,
0.8352941,
0.6156863,
0.078431375,
0.8235294,
0.59607846,
0.023529412,
0.8235294,
0.6039216,
0.019607844,
0.83137256,
0.6313726,
0.03137255,
0.827451,
0.64705884,
0.039215688,
0.8392157,
0.65882355,
0.050980393,
0.8627451,
0.6666667,
0.08235294,
0.8784314,
0.6745098,
0.101960786,
0.88235295,
0.6784314,
```

```
0.10980392,
0.87058824,
0.6666667,
0.09411765,
0.87058824,
0.6745098,
0.08235294,
0.8666667,
0.6745098,
0.07058824,
0.8509804,
0.654902,
0.0627451,
0.8509804,
0.6509804,
0.05882353,
0.84313726,
0.6313726,
0.050980393,
0.8392157,
0.62352943,
0.05490196,
0.8392157,
0.62352943,
0.05490196,
0.81960785,
0.6156863,
0.047058824,
0.80784315,
0.5921569,
0.03529412,
0.72156864,
0.5529412,
0.11372549,
0.5764706,
0.4862745,
0.2901961,
0.54509807,
0.47843137,
0.3764706,
0.32941177,
0.22745098,
0.08235294,
0.35686275,
0.24313726,
0.07058824,
0.13725491,
0.19215687,
```

```
0.0627451,
0.16470589,
0.19215687,
0.05882353,
0.2,
0.18039216,
0.023529412,
0.30588236,
0.26666668,
0.12941177,
0.21568628,
0.19215687,
0.13725491,
0.15686275,
0.101960786,
0.0,
0.42745098,
0.3019608,
0.0,
0.7764706,
0.60784316,
0.16470589,
0.827451,
0.62352943,
0.09019608,
0.827451,
0.6156863,
0.02745098,
0.81960785,
0.61960787,
0.02745098,
0.827451,
0.6313726,
0.039215688,
0.827451,
0.6509804,
0.050980393,
0.8392157,
0.6627451,
0.0627451,
0.85882354,
0.65882355,
0.06666667,
0.8627451,
0.64705884,
0.078431375,
0.85882354,
0.6431373,
```

```
0.078431375,
0.8745098,
0.67058825,
0.101960786,
0.8627451,
0.65882355,
0.078431375,
0.85882354,
0.6627451,
0.07058824,
0.8666667,
0.6666667,
0.07450981,
0.8392157,
0.6392157,
0.047058824,
0.8392157,
0.63529414,
0.0627451,
0.83137256,
0.63529414,
0.05882353,
0.827451,
0.62352943,
0.050980393,
0.79607844,
0.59607846,
0.02745098,
0.8,
0.6,
0.047058824,
0.6862745,
0.5254902,
0.10980392,
0.32941177,
0.25882354,
0.078431375,
0.2627451,
0.21568628,
0.12156863,
0.29803923,
0.20392157,
0.05490196,
0.3372549,
0.22352941,
0.050980393,
0.15294118,
0.23529412,
```

0.07450981,
0.18431373,
0.24313726,
0.07450981,
0.18039216,
0.1764706,
0.003921569,
0.28235295,
0.2627451,
0.11372549,
0.1882353,
0.16862746,
0.08235294,
0.11372549,
0.0627451,
0.0,
0.32941177,
0.22352941,
0.0,
0.7254902,
0.5764706,
0.18039216,
0.7921569,
0.6039216,
0.078431375,
0.80784315,
0.6039216,
0.023529412,
0.8117647,
0.6117647,
0.043137256,
0.8235294,
0.62352943,
0.0627451,
0.827451,
0.64705884,
0.07058824,
0.8352941,
0.654902,
0.07450981,
0.8509804,
0.654902,
0.078431375,
0.8509804,
0.6509804,
0.09019608,
0.8509804,
0.64705884,

```
0.11372549,
0.8352941,
0.6313726,
0.09803922,
0.8509804,
0.6509804,
0.09803922,
0.85490197,
0.6509804,
0.078431375,
0.85490197,
0.65882355,
0.06666667,
0.8352941,
0.6392157,
0.05490196,
0.84313726,
0.65882355,
0.101960786,
0.85490197,
0.67058825,
0.11372549,
0.8352941,
0.6313726,
0.050980393,
0.8117647,
0.6117647,
0.043137256,
0.78431374,
0.6117647,
0.08235294,
0.61960787,
0.49411765,
0.09803922,
0.19607843,
0.14901961,
0.0,
0.12156863,
0.09019608,
0.007843138,
0.2901961,
0.20392157,
0.050980393,
0.3372549,
0.21568628,
0.03529412,
0.18039216,
0.27058825,
```

```
0.09019608,
0.19215687,
0.25882354,
0.08235294,
0.1764706,
0.1882353,
0.019607844,
0.2627451,
0.24313726,
0.08627451,
0.2627451,
0.23137255,
0.08627451,
0.18039216,
0.1254902,
0.0,
0.25490198,
0.14901961,
0.0,
0.6666667,
0.5176471,
0.12941177,
0.78431374,
0.6,
0.08235294,
0.8117647,
0.6117647,
0.043137256,
0.8156863,
0.62352943,
0.078431375,
0.81960785,
0.63529414,
0.08627451,
0.827451,
0.64705884,
0.078431375,
0.8352941,
0.654902,
0.08627451,
0.85882354,
0.65882355,
0.105882354,
0.8745098,
0.67058825,
0.13725491,
0.80784315,
0.6156863,
```

```
0.10980392,
0.7254902,
0.5411765,
0.03137255,
0.827451,
0.63529414,
0.09803922,
0.8509804,
0.6627451,
0.09803922,
0.8352941,
0.6509804,
0.07058824,
0.8235294,
0.6431373,
0.06666667,
0.8235294,
0.64705884,
0.09411765,
0.8235294,
0.64705884,
0.09411765,
0.8156863,
0.6156863,
0.047058824,
0.81960785,
0.61960787,
0.05882353,
0.79607844,
0.62352943,
0.10980392,
0.5411765,
0.41568628,
0.019607844,
0.16078432,
0.09803922,
0.0,
0.12156863,
0.07058824,
0.0,
0.29803923,
0.20392157,
0.023529412,
0.3254902,
0.20784314,
0.003921569,
0.22352941,
0.29803923,
```

```
0.11764706,
0.1882353,
0.23921569,
0.07058824,
0.17254902,
0.1764706,
0.043137256,
0.28627452,
0.25882354,
0.09411765,
0.49411765,
0.4509804,
0.20784314,
0.46666667,
0.39215687,
0.09019608,
0.47843137,
0.36078432,
0.007843138,
0.69803923,
0.5372549,
0.12156863,
0.76862746,
0.5764706,
0.07450981,
0.7921569,
0.5882353,
0.05490196,
0.78431374,
0.6,
0.05882353,
0.7882353,
0.61960787,
0.06666667,
0.8039216,
0.6392157,
0.07450981,
0.81960785,
0.63529414,
0.078431375,
0.85490197,
0.6313726,
0.10980392,
0.8784314,
0.6627451,
0.16470589,
0.8352941,
0.64705884,
```

```
0.16078432,
0.7372549,
0.5647059,
0.07450981,
0.81960785,
0.654902,
0.11372549,
0.827451,
0.65882355,
0.105882354,
0.827451,
0.6431373,
0.08627451,
0.81960785,
0.6431373,
0.09019608,
0.8039216,
0.63529414,
0.09019608,
0.78431374,
0.6156863,
0.07058824,
0.827451,
0.63529414,
0.08235294,
0.80784315,
0.6039216,
0.07058824,
0.79607844,
0.6,
0.10980392,
0.5803922,
0.4117647,
0.0,
0.3137255,
0.2,
0.0,
0.34901962,
0.25490198,
0.003921569,
0.43529412,
0.3372549,
0.078431375,
0.3137255,
0.20392157,
0.0,
0.16470589,
0.23137255,
```

```
0.06666667,
0.16862746,
0.21568628,
0.06666667,
0.18431373,
0.19607843,
0.07450981,
0.2509804,
0.23529412,
0.08627451,
0.43137255,
0.39607844,
0.14117648,
0.5176471,
0.4509804,
0.13725491,
0.6,
0.49411765,
0.13725491,
0.70980394,
0.5647059,
0.15686275,
0.76862746,
0.5764706,
0.10980392,
0.7882353,
0.5921569,
0.078431375,
0.78039217,
0.6,
0.05882353,
0.7764706,
0.6117647,
0.047058824,
0.7921569,
0.63529414,
0.05882353,
0.8,
0.63529414,
0.07058824,
0.83137256,
0.61960787,
0.09411765,
0.8509804,
0.63529414,
0.12941177,
0.827451,
0.6509804,
```

```
0.14509805,
0.79607844,
0.6392157,
0.1254902,
0.8,
0.63529414,
0.09411765,
0.78039217,
0.6117647,
0.05882353,
0.80784315,
0.627451,
0.08627451,
0.80784315,
0.627451,
0.09019608,
0.80784315,
0.6431373,
0.10980392,
0.7882353,
0.61960787,
0.078431375,
0.827451,
0.6431373,
0.09411765,
0.7921569,
0.59607846,
0.06666667,
0.78039217,
0.58431375,
0.101960786,
0.7058824,
0.53333336,
0.105882354,
0.627451,
0.49019608,
0.13725491,
0.63529414,
0.52156866,
0.22352941,
0.5686275,
0.4627451,
0.2,
0.24705882,
0.14901961,
0.0,
0.15686275,
0.22352941,
```

```
0.08235294,
0.13725491,
0.2,
0.05882353,
0.15294118,
0.19607843,
0.078431375,
0.2,
0.22352941,
0.09019608,
0.26666668,
0.25490198,
0.08627451,
0.36078432,
0.31764707,
0.105882354,
0.49019608,
0.41568628,
0.14509805,
0.63529414,
0.5176471,
0.16470589,
0.7254902,
0.5686275,
0.13333334,
0.78039217,
0.6039216,
0.09803922,
0.8,
0.6156863,
0.05882353,
0.79607844,
0.62352943,
0.039215688,
0.7882353,
0.6431373,
0.05490196,
0.7921569,
0.6431373,
0.06666667,
0.8117647,
0.63529414,
0.07450981,
0.827451,
0.6431373,
0.08627451,
0.8039216,
0.6431373,
```

```
0.078431375,
0.827451,
0.6666667,
0.101960786,
0.8117647,
0.64705884,
0.08235294,
0.8156863,
0.6392157,
0.08627451,
0.8352941,
0.654902,
0.11764706,
0.83137256,
0.64705884,
0.12941177,
0.827451,
0.654902,
0.14117648,
0.8117647,
0.6392157,
0.11764706,
0.80784315,
0.6313726,
0.078431375,
0.79607844,
0.627451,
0.08627451,
0.77254903,
0.6,
0.10980392,
0.73333335,
0.5764706,
0.14117648,
0.6745098,
0.5411765,
0.1882353,
0.5882353,
0.47058824,
0.1882353,
0.4745098,
0.36862746,
0.16078432,
0.22352941,
0.11764706,
0.0,
0.14117648,
0.21960784,
```

```
0.07450981,
0.14117648,
0.21960784,
0.07450981,
0.14117648,
0.2,
0.07058824,
0.2,
0.24313726,
0.11764706,
0.23529412,
0.25882354,
0.1254902,
0.2,
0.19215687,
0.039215688,
0.29803923,
0.27058825,
0.05882353,
0.3764706,
0.31764707,
0.047058824,
0.5372549,
0.4392157,
0.10980392,
0.64705884,
0.5176471,
0.11372549,
0.72156864,
0.5647059,
0.08627451,
0.73333335,
0.5764706,
0.050980393,
0.7372549,
0.6039216,
0.078431375,
0.7607843,
0.627451,
0.105882354,
0.7882353,
0.63529414,
0.12156863,
0.80784315,
0.654902,
0.14117648,
0.83137256,
0.68235296,
```

```
0.16078432,
0.8156863,
0.6666667,
0.13725491,
0.8235294,
0.6666667,
0.15294118,
0.8156863,
0.654902,
0.15294118,
0.7921569,
0.627451,
0.15294118,
0.7921569,
0.6156863,
0.14901961,
0.78039217,
0.6,
0.12941177,
0.7764706,
0.6039216,
0.12941177,
0.7490196,
0.59607846,
0.13725491,
0.69411767,
0.56078434,
0.12941177,
0.61960787,
0.5058824,
0.11372549,
0.50980395,
0.4,
0.07058824,
0.37254903,
0.2784314,
0.003921569,
0.25882354,
0.1764706,
0.0,
0.25490198,
0.16862746,
0.023529412,
0.19215687,
0.10980392,
0.0,
0.14117648,
0.24705882,
```

```
0.08627451,
0.14901961,
0.24705882,
0.08235294,
0.1764706,
0.24705882,
0.09019608,
0.21176471,
0.27058825,
0.11764706,
0.21960784,
0.26666668,
0.1254902,
0.21176471,
0.2509804,
0.105882354,
0.21960784,
0.2627451,
0.09803922,
0.21176471,
0.23921569,
0.07450981,
0.1882353,
0.18039216,
0.03137255,
0.21960784,
0.16470589,
0.0,
0.4862745,
0.3647059,
0.050980393,
0.6627451,
0.52156866,
0.14509805,
0.65882355,
0.54509807,
0.15294118,
0.6627451,
0.5529412,
0.16078432,
0.6901961,
0.56078434,
0.18039216,
0.7176471,
0.58431375,
0.20392157,
0.72156864,
0.5803922,
```

```
0.20392157,
0.7411765,
0.6,
0.21960784,
0.7176471,
0.5882353,
0.18431373,
0.7176471,
0.58431375,
0.19215687,
0.7137255,
0.5764706,
0.20784314,
0.72156864,
0.5647059,
0.1882353,
0.7411765,
0.5568628,
0.14117648,
0.6862745,
0.52156866,
0.13725491,
0.53333336,
0.42352942,
0.13725491,
0.39215687,
0.3254902,
0.11372549,
0.32156864,
0.2784314,
0.101960786,
0.24705882,
0.22352941,
0.09019608,
0.21176471,
0.1882353,
0.08627451,
0.12156863,
0.09411765,
0.019607844,
0.16078432,
0.1254902,
0.06666667,
0.16078432,
0.1254902,
0.06666667,
0.16470589,
0.28627452,
```

```
0.12156863,
0.15686275,
0.2627451,
0.09411765,
0.19607843,
0.2784314,
0.11764706,
0.23529412,
0.29803923,
0.14509805,
0.2,
0.27058825,
0.12156863,
0.21176471,
0.28235295,
0.13333334,
0.2509804,
0.31764707,
0.1764706,
0.20392157,
0.2627451,
0.14901961,
0.11372549,
0.14117648,
0.078431375,
0.09411765,
0.06666667,
0.0,
0.29803923,
0.1882353,
0.0,
0.6039216,
0.46666667,
0.14901961,
0.41960785,
0.3137255,
0.003921569,
0.28627452,
0.19215687,
0.0,
0.38039216,
0.28627452,
0.011764706,
0.41568628,
0.30588236,
0.050980393,
0.4745098,
0.34901962,
```

```
0.11764706,
0.50980395,
0.38431373,
0.14509805,
0.4509804,
0.35686275,
0.07450981,
0.41960785,
0.3254902,
0.03529412,
0.38039216,
0.26666668,
0.0,
0.41568628,
0.27450982,
0.0,
0.65882355,
0.4862745,
0.105882354,
0.50980395,
0.3529412,
0.011764706,
0.16078432,
0.078431375,
0.0,
0.15294118,
0.12941177,
0.03529412,
0.18431373,
0.18039216,
0.10980392,
0.24313726,
0.2509804,
0.2,
0.25882354,
0.25882354,
0.22745098,
0.13725491,
0.1254902,
0.09803922,
0.13333334,
0.11764706,
0.07450981,
0.13725491,
0.11372549,
0.06666667,
0.15686275,
0.26666668,
```

```
0.10980392,
0.17254902,
0.27450982,
0.12156863,
0.19607843,
0.2784314,
0.14901961,
0.19607843,
0.28235295,
0.14117648,
0.16862746,
0.26666668,
0.101960786,
0.17254902,
0.2627451,
0.09803922,
0.19215687,
0.25882354,
0.11764706,
0.1882353,
0.23137255,
0.11372549,
0.15294118,
0.16078432,
0.07450981,
0.09803922,
0.050980393,
0.0,
0.3372549,
0.21176471,
0.0,
0.6431373,
0.49411765,
0.16862746,
0.38431373,
0.27058825,
0.011764706,
0.18039216,
0.09019608,
0.0,
0.21176471,
0.14117648,
0.0,
0.2509804,
0.17254902,
0.03529412,
0.26666668,
0.15686275,
```

```
0.07058824,
0.28235295,
0.18039216,
0.09019608,
0.20784314,
0.14901961,
0.0,
0.21568628,
0.16078432,
0.0,
0.19607843,
0.11372549,
0.0,
0.2784314,
0.16078432,
0.0,
0.65882355,
0.49803922,
0.14509805,
0.5058824,
0.3647059,
0.02745098,
0.11764706,
0.05882353,
0.0,
0.08235294,
0.07450981,
0.0,
0.078431375,
0.07450981,
0.0,
0.08235294,
0.07058824,
0.0,
0.1254902,
0.09803922,
0.02745098,
0.14509805,
0.11372549,
0.039215688,
0.1764706,
0.13333334,
0.0627451,
0.13333334,
0.09019608,
0.011764706,
0.1764706,
0.28235295,
```

```
0.12156863,
0.1764706,
0.2784314,
0.1254902,
0.18039216,
0.27450982,
0.14901961,
0.17254902,
0.26666668,
0.13333334,
0.15294118,
0.26666668,
0.09411765,
0.1764706,
0.2784314,
0.101960786,
0.22352941,
0.28627452,
0.14117648,
0.23529412,
0.27450982,
0.14117648,
0.22352941,
0.22745098,
0.105882354,
0.23529412,
0.18431373,
0.011764706,
0.5019608,
0.38431373,
0.101960786,
0.6901961,
0.54509807,
0.23921569,
0.5647059,
0.44705883,
0.20392157,
0.45882353,
0.36862746,
0.16470589,
0.4392157,
0.35686275,
0.17254902,
0.45882353,
0.37254903,
0.21960784,
0.49019608,
0.37254903,
```

```
0.27058825,
0.46666667,
0.36078432,
0.25490198,
0.39607844,
0.3254902,
0.16862746,
0.40784314,
0.3529412,
0.16862746,
0.4,
0.31764707,
0.13333334,
0.47058824,
0.36862746,
0.12941177,
0.6784314,
0.5372549,
0.19215687,
0.58431375,
0.45882353,
0.13333334,
0.25490198,
0.2,
0.015686275,
0.16078432,
0.14117648,
0.02745098,
0.1882353,
0.16862746,
0.05490196,
0.22352941,
0.19607843,
0.09411765,
0.25490198,
0.21568628,
0.11764706,
0.25882354,
0.21568628,
0.12941177,
0.18431373,
0.14117648,
0.05490196,
0.11764706,
0.08627451,
0.0,
0.16470589,
0.24705882,
```

```
0.08627451,
0.18039216,
0.26666668,
0.11372549,
0.15294118,
0.24705882,
0.105882354,
0.13725491,
0.24705882,
0.09803922,
0.17254902,
0.28235295,
0.1254902,
0.18431373,
0.28235295,
0.11764706,
0.1882353,
0.25490198,
0.09019608,
0.23137255,
0.2627451,
0.101960786,
0.2627451,
0.27058825,
0.11764706,
0.3254902,
0.29411766,
0.11372549,
0.49019608,
0.39607844,
0.16078432,
0.5803922,
0.45882353,
0.20392157,
0.6156863,
0.49803922,
0.24705882,
0.627451,
0.52156866,
0.25882354,
0.6666667,
0.54901963,
0.2627451,
0.6745098,
0.54901963,
0.25490198,
0.69803923,
0.5647059,
```

```
0.28235295,
0.7019608,
0.5686275,
0.2901961,
0.6745098,
0.5647059,
0.2784314,
0.6666667,
0.5647059,
0.27058825,
0.654902,
0.5529412,
0.2509804,
0.7019608,
0.59607846,
0.2784314,
0.69411767,
0.5686275,
0.24313726,
0.627451,
0.5137255,
0.21568628,
0.44313726,
0.38039216,
0.14901961,
0.36862746,
0.32156864,
0.13333334,
0.3647059,
0.3137255,
0.14901961,
0.3529412,
0.3019608,
0.16862746,
0.30588236,
0.26666668,
0.16078432,
0.21568628,
0.19215687,
0.105882354,
0.09019608,
0.078431375,
0.003921569,
0.09411765,
0.09019608,
0.011764706,
0.1882353,
0.25882354,
```

```
0.101960786,
0.16470589,
0.24705882,
0.09411765,
0.2,
0.3019608,
0.15686275,
0.20392157,
0.3137255,
0.16470589,
0.14901961,
0.25490198,
0.11764706,
0.15294118,
0.24705882,
0.105882354,
0.17254902,
0.22745098,
0.09019608,
0.14117648,
0.16862746,
0.03529412,
0.11764706,
0.12156863,
0.0,
0.13725491,
0.11372549,
0.0,
0.19607843,
0.1254902,
0.0,
0.24705882,
0.15686275,
0.0,
0.27450982,
0.1882353,
0.0,
0.3137255,
0.23529412,
0.003921569,
0.35686275,
0.26666668,
0.0,
0.37254903,
0.2784314,
0.0,
0.40784314,
0.29803923,
```

```
0.011764706,
0.40784314,
0.29411766,
0.019607844,
0.41960785,
0.32156864,
0.05882353,
0.38431373,
0.30588236,
0.03529412,
0.37254903,
0.3019608,
0.019607844,
0.38431373,
0.30980393,
0.039215688,
0.35686275,
0.2627451,
0.019607844,
0.3137255,
0.23137255,
0.007843138,
0.25882354,
0.20392157,
0.011764706,
0.23921569,
0.19215687,
0.02745098,
0.21568628,
0.16078432,
0.015686275,
0.1764706,
0.12156863,
0.007843138,
0.12156863,
0.09803922,
0.011764706,
0.09803922,
0.08627451,
0.019607844,
0.09019608,
0.09411765,
0.039215688,
0.08627451,
0.09411765,
0.039215688,
0.1764706,
0.25490198,
```

```
0.10980392,
0.18431373,
0.27058825,
0.12941177,
0.21176471,
0.30588236,
0.17254902,
0.18039216,
0.2784314,
0.15294118,
0.12941177,
0.21960784,
0.105882354,
0.14117648,
0.21176471,
0.10980392,
0.14117648,
0.1882353,
0.101960786,
0.11372549,
0.12941177,
0.0627451,
0.078431375,
0.08235294,
0.02745098,
0.05490196,
0.039215688,
0.0,
0.08627451,
0.05490196,
0.003921569,
0.09803922,
0.0627451,
0.0,
0.08235294,
0.05882353,
0.0,
0.09803922,
0.08235294,
0.0,
0.07450981,
0.078431375,
0.0,
0.09803922,
0.09411765,
0.0,
0.11372549,
0.08627451,
```

```
0.0,
0.101960786,
0.07450981,
0.0,
0.10980392,
0.09411765,
0.0,
0.09411765,
0.09019608,
0.0,
0.101960786,
0.08627451,
0.0,
0.09019608,
0.06666667,
0.0,
0.10980392,
0.07058824,
0.0,
0.105882354,
0.07450981,
0.0,
0.105882354,
0.078431375,
0.007843138,
0.08627451,
0.0627451,
0.007843138,
0.09411765,
0.0627451,
0.011764706,
0.09803922,
0.07058824,
0.03137255,
0.07450981,
0.0627451,
0.03529412,
0.078431375,
0.07450981,
0.05490196,
0.105882354,
0.10980392,
0.078431375,
0.06666667,
0.078431375,
0.043137256,
0.15686275,
0.23529412,
```

```
0.09803922,
0.16470589,
0.2509804,
0.10980392,
0.18039216,
0.27450982,
0.14901961,
0.14509805,
0.23529412,
0.12156863,
0.1254902,
0.20392157,
0.09803922,
0.15294118,
0.21176471,
0.12941177,
0.1254902,
0.15686275,
0.105882354,
0.07058824,
0.08235294,
0.05490196,
0.07058824,
0.06666667,
0.050980393,
0.078431375,
0.0627451,
0.05882353,
0.08627451,
0.07058824,
0.07450981,
0.07450981,
0.07058824,
0.0627451,
0.078431375,
0.08235294,
0.050980393,
0.07450981,
0.09803922,
0.050980393,
0.06666667,
0.11372549,
0.06666667,
0.06666667,
0.105882354,
0.0627451,
0.0627451,
0.07058824,
```

0.0627451]]}},
'label_xf': {'floatList': {'value': [0.0,
1.0,

```
0.0,
0.0,
0.0,
0.0,
0.0,
0.0,
0.0,
0.0]}}}}}]
```

1.5.7 Wrap Up

This notebook demonstrates how to do feature engineering with image datasets as opposed to simple tabular data. This should come in handy in your computer vision projects and you can also try replicating this process with other image datasets from TFDS.