NN

April 1, 2023

[1]:
```
import matplotlib
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import seaborn as sns
import statsmodels.api as sm

%matplotlib inline
```

[2]:
```
df_out = pd.read_pickle('df_out.pkl')
df_breeds = pd.read_pickle('df_breeds.pkl')
df_out_with_breeds_info = pd.read_pickle('df_out_with_breeks_info.pkl')
df_breeds_with_info = pd.read_pickle('df_breeds_with_info.pkl')
df_out.info()
df_out.head()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 149511 entries, 0 to 149510
Data columns (total 32 columns):
 #   Column                   Non-Null Count   Dtype
---  ------                   --------------   -----
 0   Animal ID                149511 non-null  string
 1   Name                     106260 non-null  string
 2   DateTime                 149511 non-null  datetime64[ns]
 3   MonthYear                149511 non-null  string
 4   Date of Birth            149511 non-null  datetime64[ns]
 5   Outcome Type             149485 non-null  string
 6   Outcome Subtype          68443 non-null   string
 7   Animal Type              149511 non-null  string
 8   Sex upon Outcome         149509 non-null  string
 9   Age upon Outcome         149465 non-null  string
 10  Breed                    149511 non-null  string
 11  Color                    149511 non-null  string
 12  Colors (count)           149511 non-null  Int64
 13  Color 0                  149511 non-null  string
 14  Color 1                  79869 non-null   string
 15  Color 0 R                135638 non-null  Float64
 16  Color 0 G                135638 non-null  Float64
```

```
 17  Color 0 B                 135638 non-null  Float64
 18  Color 0 H                 135638 non-null  Float64
 19  Color 0 S                 135638 non-null  Float64
 20  Color 0 V                 135638 non-null  Float64
 21  Color 1 R                 78596 non-null   Float64
 22  Color 1 G                 78596 non-null   Float64
 23  Color 1 B                 78596 non-null   Float64
 24  Color 1 H                 78596 non-null   Float64
 25  Color 1 S                 78596 non-null   Float64
 26  Color 1 V                 78596 non-null   Float64
 27  Age upon Outcome (years)  149465 non-null  Float64
 28  Male                      149509 non-null  boolean
 29  Female                    149509 non-null  boolean
 30  NeuteredOrSpayed          149509 non-null  boolean
 31  Adopted                   149485 non-null  boolean
dtypes: Float64(13), Int64(1), boolean(4), datetime64[ns](2), string(12)
memory usage: 35.1 MB
```

[2]:

	Animal ID	Name	DateTime	MonthYear	Date of Birth	Outcome Type	\
0	A794011	Chunk	2019-05-08 18:20:00	May 2019	2017-05-02	Rto-Adopt	
1	A776359	Gizmo	2018-07-18 16:02:00	Jul 2018	2017-07-12	Adoption	
2	A821648	<NA>	2020-08-16 11:38:00	Aug 2020	2019-08-16	Euthanasia	
3	A720371	Moose	2016-02-13 17:59:00	Feb 2016	2015-10-08	Adoption	
4	A674754	<NA>	2014-03-18 11:47:00	Mar 2014	2014-03-12	Transfer	

	Outcome Subtype	Animal Type	Sex upon Outcome	Age upon Outcome	…	\
0	<NA>	Cat	Neutered Male	2 years	…	
1	<NA>	Dog	Neutered Male	1 year	…	
2	<NA>	Other	Unknown	1 year	…	
3	<NA>	Dog	Neutered Male	4 months	…	
4	Partner	Cat	Intact Male	6 days	…	

	Color 1 G	Color 1 B	Color 1 H	Color 1 S	Color 1 V	\
0	1.0	1.0	0.0	0.0	1.0	
1	0.44	0.09	0.119444	0.85	0.59	
2	<NA>	<NA>	<NA>	<NA>	<NA>	
3	<NA>	<NA>	<NA>	<NA>	<NA>	
4	<NA>	<NA>	<NA>	<NA>	<NA>	

	Age upon Outcome (years)	Male	Female	NeuteredOrSpayed	Adopted
0	2.0	True	False	True	True
1	1.0	True	False	True	True
2	1.0	False	False	False	False
3	0.333333	True	False	True	True
4	0.016438	True	False	False	False

[5 rows x 32 columns]

1 NN

This section will attempt to look for correlations between the names of individual animals, the names and characteristics of their breeds, their ages, and whether or not they got adopted.

It uses a neural network with supervised learning to predict the value of "Adopted" given the other features.

To regress text against numeric values, the vector embedding of the string is used. These embeddings are computed beforehand to optimize training.

1.1 Loading and experimenting with word embeddings

[3]:
```
%pip install tensorflow
%pip install tensorflow_hub

# # NOTE: when I ran this, it said:
# # "Note: you may need to restart the kernel to use updated packages."
```

```
Requirement already satisfied: tensorflow in
/home/isaac/miniconda3/envs/cse3380/lib/python3.10/site-packages (2.12.0)
Requirement already satisfied: gast<=0.4.0,>=0.2.1 in
/home/isaac/miniconda3/envs/cse3380/lib/python3.10/site-packages (from
tensorflow) (0.4.0)
Requirement already satisfied: astunparse>=1.6.0 in
/home/isaac/miniconda3/envs/cse3380/lib/python3.10/site-packages (from
tensorflow) (1.6.3)
Requirement already satisfied: grpcio<2.0,>=1.24.3 in
/home/isaac/miniconda3/envs/cse3380/lib/python3.10/site-packages (from
tensorflow) (1.53.0)
Requirement already satisfied: libclang>=13.0.0 in
/home/isaac/miniconda3/envs/cse3380/lib/python3.10/site-packages (from
tensorflow) (16.0.0)
Requirement already satisfied: flatbuffers>=2.0 in
/home/isaac/miniconda3/envs/cse3380/lib/python3.10/site-packages (from
tensorflow) (23.3.3)
Requirement already satisfied: keras<2.13,>=2.12.0 in
/home/isaac/miniconda3/envs/cse3380/lib/python3.10/site-packages (from
tensorflow) (2.12.0)
Requirement already satisfied: tensorflow-estimator<2.13,>=2.12.0 in
/home/isaac/miniconda3/envs/cse3380/lib/python3.10/site-packages (from
tensorflow) (2.12.0)
Requirement already satisfied: h5py>=2.9.0 in
/home/isaac/miniconda3/envs/cse3380/lib/python3.10/site-packages (from
tensorflow) (3.8.0)
Requirement already satisfied: absl-py>=1.0.0 in
/home/isaac/miniconda3/envs/cse3380/lib/python3.10/site-packages (from
tensorflow) (1.4.0)
Requirement already satisfied: jax>=0.3.15 in
```

```
/home/isaac/miniconda3/envs/cse3380/lib/python3.10/site-packages (from
tensorflow) (0.4.7)
Requirement already satisfied: google-pasta>=0.1.1 in
/home/isaac/miniconda3/envs/cse3380/lib/python3.10/site-packages (from
tensorflow) (0.2.0)
Requirement already satisfied: packaging in
/home/isaac/miniconda3/envs/cse3380/lib/python3.10/site-packages (from
tensorflow) (21.3)
Requirement already satisfied: typing-extensions>=3.6.6 in
/home/isaac/miniconda3/envs/cse3380/lib/python3.10/site-packages (from
tensorflow) (4.3.0)
Requirement already satisfied: opt-einsum>=2.3.2 in
/home/isaac/miniconda3/envs/cse3380/lib/python3.10/site-packages (from
tensorflow) (3.3.0)
Requirement already satisfied: tensorboard<2.13,>=2.12 in
/home/isaac/miniconda3/envs/cse3380/lib/python3.10/site-packages (from
tensorflow) (2.12.0)
Requirement already satisfied: six>=1.12.0 in
/home/isaac/miniconda3/envs/cse3380/lib/python3.10/site-packages (from
tensorflow) (1.16.0)
Requirement already satisfied: wrapt<1.15,>=1.11.0 in
/home/isaac/miniconda3/envs/cse3380/lib/python3.10/site-packages (from
tensorflow) (1.14.1)
Requirement already satisfied: tensorflow-io-gcs-filesystem>=0.23.1 in
/home/isaac/miniconda3/envs/cse3380/lib/python3.10/site-packages (from
tensorflow) (0.31.0)
Requirement already satisfied:
protobuf!=4.21.0,!=4.21.1,!=4.21.2,!=4.21.3,!=4.21.4,!=4.21.5,<5.0.0dev,>=3.20.3
in /home/isaac/miniconda3/envs/cse3380/lib/python3.10/site-packages (from
tensorflow) (4.22.1)
Requirement already satisfied: setuptools in
/home/isaac/miniconda3/envs/cse3380/lib/python3.10/site-packages (from
tensorflow) (63.4.1)
Requirement already satisfied: termcolor>=1.1.0 in
/home/isaac/miniconda3/envs/cse3380/lib/python3.10/site-packages (from
tensorflow) (2.2.0)
Requirement already satisfied: numpy<1.24,>=1.22 in
/home/isaac/miniconda3/envs/cse3380/lib/python3.10/site-packages (from
tensorflow) (1.23.1)
Requirement already satisfied: wheel<1.0,>=0.23.0 in
/home/isaac/miniconda3/envs/cse3380/lib/python3.10/site-packages (from
astunparse>=1.6.0->tensorflow) (0.37.1)
Requirement already satisfied: ml-dtypes>=0.0.3 in
/home/isaac/miniconda3/envs/cse3380/lib/python3.10/site-packages (from
jax>=0.3.15->tensorflow) (0.0.4)
Requirement already satisfied: scipy>=1.7 in
/home/isaac/miniconda3/envs/cse3380/lib/python3.10/site-packages (from
jax>=0.3.15->tensorflow) (1.9.1)
```

```
Requirement already satisfied: requests<3,>=2.21.0 in
/home/isaac/miniconda3/envs/cse3380/lib/python3.10/site-packages (from
tensorboard<2.13,>=2.12->tensorflow) (2.28.1)
Requirement already satisfied: google-auth<3,>=1.6.3 in
/home/isaac/miniconda3/envs/cse3380/lib/python3.10/site-packages (from
tensorboard<2.13,>=2.12->tensorflow) (2.17.0)
Requirement already satisfied: google-auth-oauthlib<0.5,>=0.4.1 in
/home/isaac/miniconda3/envs/cse3380/lib/python3.10/site-packages (from
tensorboard<2.13,>=2.12->tensorflow) (0.4.6)
Requirement already satisfied: tensorboard-plugin-wit>=1.6.0 in
/home/isaac/miniconda3/envs/cse3380/lib/python3.10/site-packages (from
tensorboard<2.13,>=2.12->tensorflow) (1.8.1)
Requirement already satisfied: tensorboard-data-server<0.8.0,>=0.7.0 in
/home/isaac/miniconda3/envs/cse3380/lib/python3.10/site-packages (from
tensorboard<2.13,>=2.12->tensorflow) (0.7.0)
Requirement already satisfied: markdown>=2.6.8 in
/home/isaac/miniconda3/envs/cse3380/lib/python3.10/site-packages (from
tensorboard<2.13,>=2.12->tensorflow) (3.4.3)
Requirement already satisfied: werkzeug>=1.0.1 in
/home/isaac/miniconda3/envs/cse3380/lib/python3.10/site-packages (from
tensorboard<2.13,>=2.12->tensorflow) (2.2.3)
Requirement already satisfied: pyparsing!=3.0.5,>=2.0.2 in
/home/isaac/miniconda3/envs/cse3380/lib/python3.10/site-packages (from
packaging->tensorflow) (3.0.9)
Requirement already satisfied: cachetools<6.0,>=2.0.0 in
/home/isaac/miniconda3/envs/cse3380/lib/python3.10/site-packages (from google-
auth<3,>=1.6.3->tensorboard<2.13,>=2.12->tensorflow) (5.3.0)
Requirement already satisfied: rsa<5,>=3.1.4 in
/home/isaac/miniconda3/envs/cse3380/lib/python3.10/site-packages (from google-
auth<3,>=1.6.3->tensorboard<2.13,>=2.12->tensorflow) (4.9)
Requirement already satisfied: pyasn1-modules>=0.2.1 in
/home/isaac/miniconda3/envs/cse3380/lib/python3.10/site-packages (from google-
auth<3,>=1.6.3->tensorboard<2.13,>=2.12->tensorflow) (0.2.8)
Requirement already satisfied: requests-oauthlib>=0.7.0 in
/home/isaac/miniconda3/envs/cse3380/lib/python3.10/site-packages (from google-
auth-oauthlib<0.5,>=0.4.1->tensorboard<2.13,>=2.12->tensorflow) (1.3.1)
Requirement already satisfied: idna<4,>=2.5 in
/home/isaac/miniconda3/envs/cse3380/lib/python3.10/site-packages (from
requests<3,>=2.21.0->tensorboard<2.13,>=2.12->tensorflow) (3.4)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in
/home/isaac/miniconda3/envs/cse3380/lib/python3.10/site-packages (from
requests<3,>=2.21.0->tensorboard<2.13,>=2.12->tensorflow) (1.26.12)
Requirement already satisfied: certifi>=2017.4.17 in
/home/isaac/miniconda3/envs/cse3380/lib/python3.10/site-packages (from
requests<3,>=2.21.0->tensorboard<2.13,>=2.12->tensorflow) (2022.12.7)
Requirement already satisfied: charset-normalizer<3,>=2 in
/home/isaac/miniconda3/envs/cse3380/lib/python3.10/site-packages (from
requests<3,>=2.21.0->tensorboard<2.13,>=2.12->tensorflow) (2.0.4)
```

```
Requirement already satisfied: MarkupSafe>=2.1.1 in
/home/isaac/miniconda3/envs/cse3380/lib/python3.10/site-packages (from
werkzeug>=1.0.1->tensorboard<2.13,>=2.12->tensorflow) (2.1.1)
Requirement already satisfied: pyasn1<0.5.0,>=0.4.6 in
/home/isaac/miniconda3/envs/cse3380/lib/python3.10/site-packages (from
pyasn1-modules>=0.2.1->google-
auth<3,>=1.6.3->tensorboard<2.13,>=2.12->tensorflow) (0.4.8)
Requirement already satisfied: oauthlib>=3.0.0 in
/home/isaac/miniconda3/envs/cse3380/lib/python3.10/site-packages (from requests-
oauthlib>=0.7.0->google-auth-
oauthlib<0.5,>=0.4.1->tensorboard<2.13,>=2.12->tensorflow) (3.2.2)
Note: you may need to restart the kernel to use updated packages.
Requirement already satisfied: tensorflow_hub in
/home/isaac/miniconda3/envs/cse3380/lib/python3.10/site-packages (0.13.0)
Requirement already satisfied: protobuf>=3.19.6 in
/home/isaac/miniconda3/envs/cse3380/lib/python3.10/site-packages (from
tensorflow_hub) (4.22.1)
Requirement already satisfied: numpy>=1.12.0 in
/home/isaac/miniconda3/envs/cse3380/lib/python3.10/site-packages (from
tensorflow_hub) (1.23.1)
Note: you may need to restart the kernel to use updated packages.
```

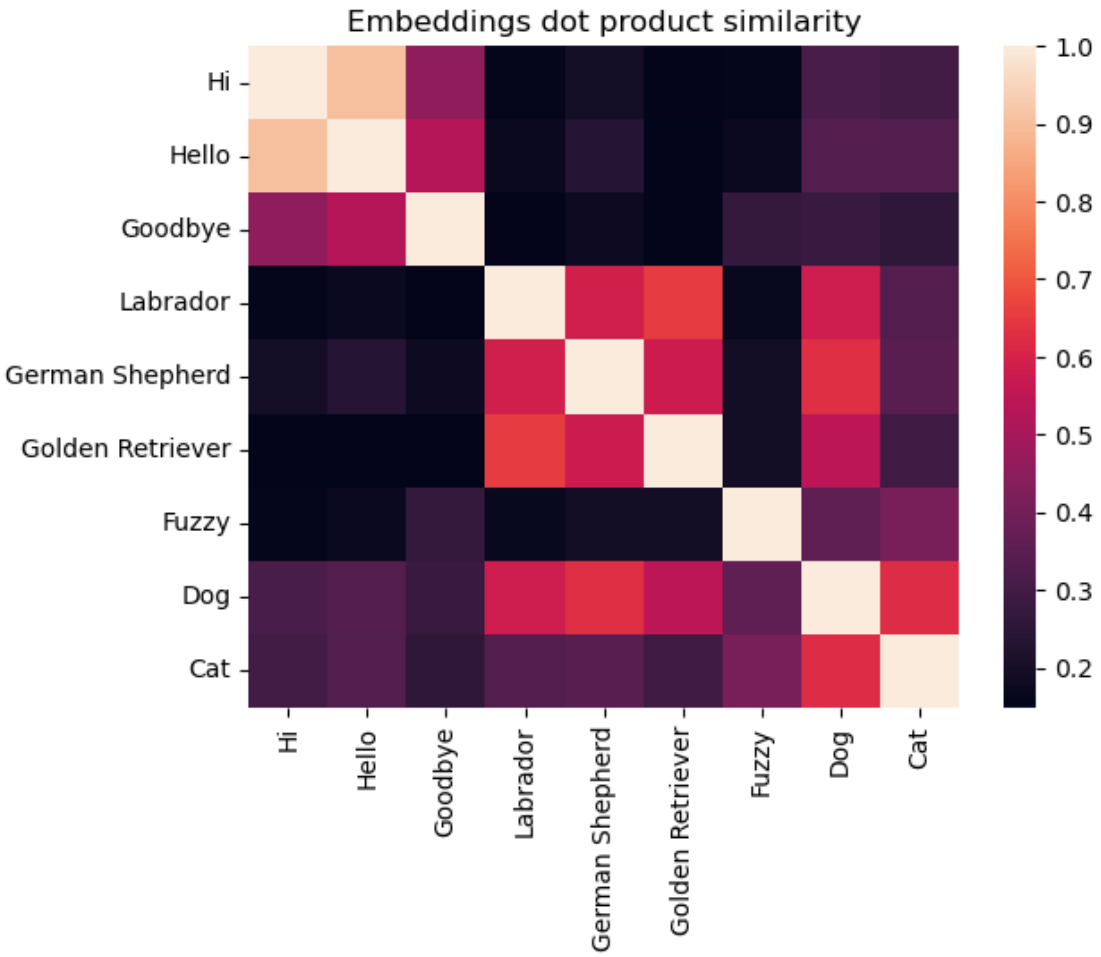
[4]:
```
# adapted from TensorFlow tutorial
# https://www.tensorflow.org/hub/tutorials/
 ↪semantic_similarity_with_tf_hub_universal_encoder
# NOTE: this model is about 1 GB, and it sometimes downloads the model again␣
 ↪even after it's
# already run and downloaded the model before

import tensorflow as tf
import tensorflow_hub as hub
import os
import re

module_url = "https://tfhub.dev/google/universal-sentence-encoder/4"
model = hub.load(module_url)
print ("module %s loaded" % module_url)
def embed(input):
  return model(input)
```

```
2023-04-01 16:59:12.819604: I tensorflow/tsl/cuda/cudart_stub.cc:28] Could not
find cuda drivers on your machine, GPU will not be used.
2023-04-01 16:59:16.418779: I tensorflow/tsl/cuda/cudart_stub.cc:28] Could not
find cuda drivers on your machine, GPU will not be used.
2023-04-01 16:59:16.440053: I tensorflow/core/platform/cpu_feature_guard.cc:182]
This TensorFlow binary is optimized to use available CPU instructions in
performance-critical operations.
To enable the following instructions: AVX2 FMA, in other operations, rebuild
```

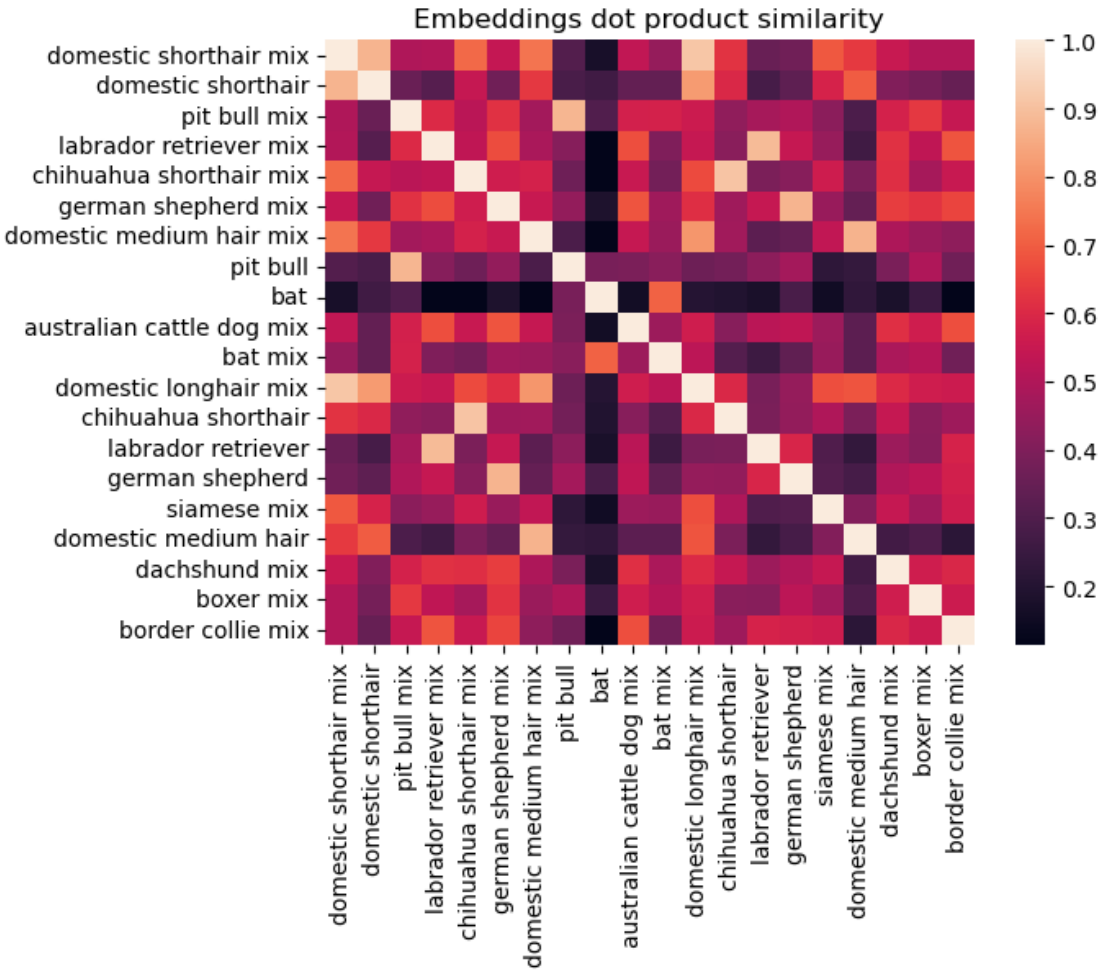
```
TensorFlow with the appropriate compiler flags.
2023-04-01 16:59:21.671574: W
tensorflow/compiler/tf2tensorrt/utils/py_utils.cc:38] TF-TRT Warning: Could not
find TensorRT

module https://tfhub.dev/google/universal-sentence-encoder/4 loaded
```

[5]:
```
# from TensorFlow example

def similarity(messages):
    embeddngs = embed(messages)
    corr = np.inner(embeddngs, embeddngs)
    g = sns.heatmap(corr, xticklabels=messages, yticklabels=messages)
    g.set_xticklabels(messages, rotation=90)
    g.set_title('Embeddings dot product similarity')

similarity([
    'Hi', 'Hello', 'Goodbye',
    'Labrador', 'German Shepherd', 'Golden Retriever',
    'Fuzzy', 'Dog', 'Cat'
])
```

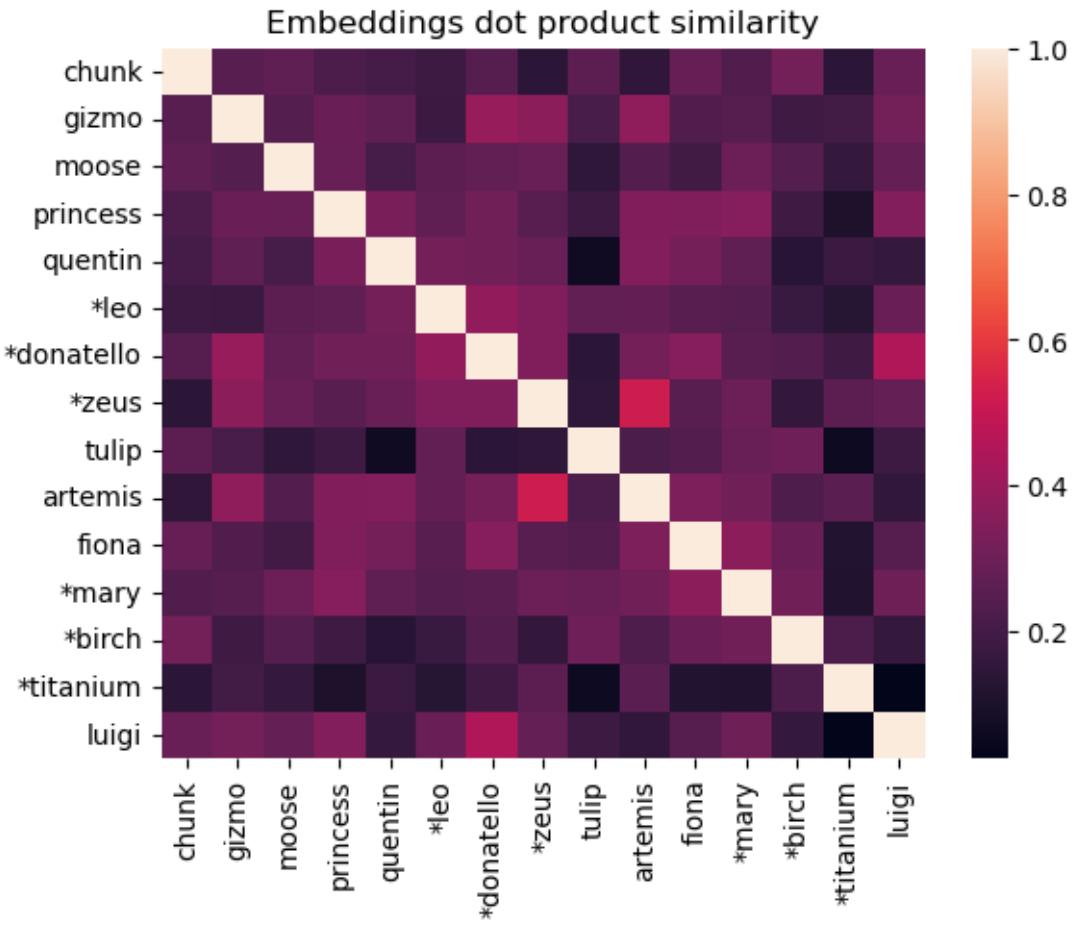
```
2023-04-01 16:59:41.537272: I tensorflow/core/common_runtime/executor.cc:1197]
[/device:CPU:0] (DEBUG INFO) Executor start aborting (this does not indicate an
error and you can ignore this message): INVALID_ARGUMENT: You must feed a value
for placeholder tensor 'inputs' with dtype string
         [[{{node inputs}}]]
```



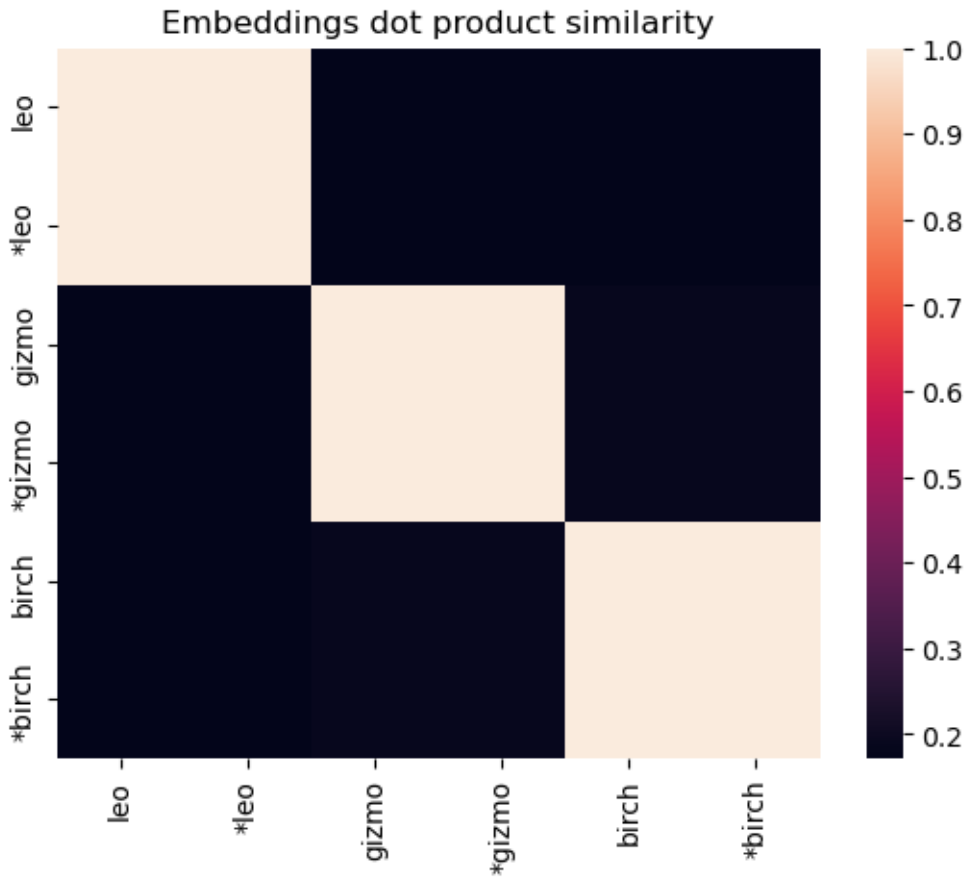
[6]:
```
similarity(list(df_breeds.Breed.str.lower().head(20)))
```



[7]:
```
similarity(list(df_out.Name.str.lower().head(20).dropna()))
```



[8]:
```
# this is good; it shows that the source dataset will regress similarly
# with or without the asterisk in front of some names

similarity([
    'leo', '*leo',
    'gizmo', '*gizmo',
    'birch', '*birch',
])
```



1.2 Regressing features to adoption

[9]:
```
from math import floor

columns_remove_groups = [
    # group 1
    [
        'Animal ID',
        'MonthYear',
        'DateTime',
        'Date of Birth',

        'Outcome Type',
        'Outcome Subtype',
        'Sex upon Outcome',
        'Age upon Outcome', # the numeric column "Age upon Outcome (years)" is␣
 ↪still kept
        'BreedsInfoName',
        'Breed (catalog)'
```

```
    ],

    # group 2
    [
        'Color 0',
        'Color 1'
    ],

    # group 3
    [
        'Breed Group AKC',
        'Breed Group CKC',
        'Breed Group UKC',
        'CKC Subgroup'
    ],
]

columns_remove = columns_remove_groups[0] + columns_remove_groups[1]

def convert_series_to_tensor(series):
    if series.dtype == 'string':
        return embed(input=list(series))
    else:
        t = tf.cast(tf.convert_to_tensor(series.astype(dtype=float)), tf.
 ↪float32)
        return tf.reshape(t, t.shape + [1])

def convert_df_tf(df):
    tensors = [convert_series_to_tensor(df[column]) for column in df.columns]
    return tf.concat(tensors, axis=1)

def network_data(df):
    df = df.drop(columns=columns_remove)
    df = df.dropna()

    X = df
    y = X.pop('Adopted')

    X_t = convert_df_tf(X)
    y_t = convert_series_to_tensor(y)

    return (X_t, y_t)

def split_df(df, ratio=0.8):
    df = df.sample(frac=1,random_state=1).reset_index()
    size_1 = floor(df.shape[0] * ratio)
    return (df.head(size_1), df.tail(df.shape[0] - size_1))
```

```
# data_size = 50000
data_size = 100000
(df_train, df_test) = split_df(df_out_with_breeds_info.head(data_size))

(X_train, y_train) = network_data(df_train)
(X_test, y_test) = network_data(df_test)
```

```
2023-04-01 16:59:46.448616: W tensorflow/tsl/framework/cpu_allocator_impl.cc:83]
Allocation of 332108800 exceeds 10% of free system memory.
2023-04-01 16:59:48.017313: W tensorflow/tsl/framework/cpu_allocator_impl.cc:83]
Allocation of 278656000 exceeds 10% of free system memory.
2023-04-01 16:59:48.878726: W tensorflow/tsl/framework/cpu_allocator_impl.cc:83]
Allocation of 497559040 exceeds 10% of free system memory.
2023-04-01 16:59:49.815487: W tensorflow/tsl/framework/cpu_allocator_impl.cc:83]
Allocation of 246758400 exceeds 10% of free system memory.
2023-04-01 16:59:50.783693: W tensorflow/tsl/framework/cpu_allocator_impl.cc:83]
Allocation of 369909760 exceeds 10% of free system memory.
```

1.3 Multilayer perceptron

[10]:
```
model1 = tf.keras.Sequential([
    tf.keras.layers.Dense(128, activation='linear'),

    tf.keras.layers.Dense(32, activation='linear'),
    tf.keras.layers.Dense(32, activation='linear'),
    tf.keras.layers.Dense(32, activation='linear'),
    tf.keras.layers.Dense(32, activation='linear'),

    tf.keras.layers.Dense(8, activation='linear'),

    tf.keras.layers.Dense(1)
])

model1.compile(
    optimizer=tf.keras.optimizers.Adam(learning_rate=0.01),
    loss=tf.losses.BinaryCrossentropy(from_logits=True),
    metrics=['accuracy']
)
```

[11]:
```
model2 = tf.keras.Sequential([
    tf.keras.layers.Dense(128, activation='linear'),

    # tf.keras.layers.Dense(32, activation='linear'),
    # tf.keras.layers.Dense(32, activation='linear'),
    # tf.keras.layers.Dense(32, activation='linear'),
    # tf.keras.layers.Dense(32, activation='linear'),
```

```
    tf.keras.layers.Dense(8, activation='linear'),

    tf.keras.layers.Dense(1)
])

model2.compile(
    optimizer=tf.keras.optimizers.Adam(learning_rate=0.01),
    loss=tf.losses.BinaryCrossentropy(from_logits=True),
    metrics=['accuracy']
)
```

[12]:
```
model3 = tf.keras.Sequential([
    tf.keras.layers.Dense(128, activation='linear'),

    tf.keras.layers.Dense(32, activation='linear'),
    tf.keras.layers.Dense(32, activation='linear'),
    tf.keras.layers.Dense(32, activation='linear'),
    tf.keras.layers.Dense(32, activation='linear'),

    tf.keras.layers.Dense(8, activation='linear'),

    tf.keras.layers.Dense(1)
])

model3.compile(
    optimizer=tf.keras.optimizers.Adam(learning_rate=0.01),
    loss=tf.losses.BinaryCrossentropy(from_logits=False),
    metrics=['accuracy']
)
```

[13]:
```
model4 = tf.keras.Sequential([
    tf.keras.layers.Dense(128, activation='linear'),

    # tf.keras.layers.Dense(32, activation='linear'),
    # tf.keras.layers.Dense(32, activation='linear'),
    # tf.keras.layers.Dense(32, activation='linear'),
    # tf.keras.layers.Dense(32, activation='linear'),

    tf.keras.layers.Dense(8, activation='linear'),

    tf.keras.layers.Dense(1)
])

model4.compile(
    optimizer=tf.keras.optimizers.Adam(learning_rate=0.01),
    loss=tf.losses.BinaryCrossentropy(from_logits=False),
```

```
    metrics=['accuracy']
)
```

[22]:
```
model5 = tf.keras.Sequential([
    tf.keras.layers.Dense(128, activation='linear'),

    tf.keras.layers.Dense(32, activation='linear'),
    tf.keras.layers.Dense(32, activation='linear'),
    tf.keras.layers.Dense(32, activation='linear'),
    tf.keras.layers.Dense(32, activation='linear'),

    tf.keras.layers.Dense(8, activation='linear'),

    tf.keras.layers.Dense(1)
])

model5.compile(
    optimizer=tf.keras.optimizers.Adam(learning_rate=0.01),
    loss=tf.losses.MeanSquaredError(),
    metrics=['accuracy']
)
```

[23]:
```
model6 = tf.keras.Sequential([
    tf.keras.layers.Dense(128, activation='linear'),

    # tf.keras.layers.Dense(32, activation='linear'),
    # tf.keras.layers.Dense(32, activation='linear'),
    # tf.keras.layers.Dense(32, activation='linear'),
    # tf.keras.layers.Dense(32, activation='linear'),

    tf.keras.layers.Dense(8, activation='linear'),

    tf.keras.layers.Dense(1)
])

model6.compile(
    optimizer=tf.keras.optimizers.Adam(learning_rate=0.01),
    loss=tf.losses.MeanSquaredError(),
    metrics=['accuracy']
)
```

[24]:
```
models = [model1, model2, model3, model4, model5, model6]
```

[25]:
```
for model in models:
    model.fit(X_train, y_train, epochs=10)
```

```
Epoch 1/10
```

```
942/942 [==============================] - 7s 8ms/step - loss: 0.7607 -
accuracy: 0.5583
Epoch 2/10
942/942 [==============================] - 7s 7ms/step - loss: 0.7402 -
accuracy: 0.5596
Epoch 3/10
942/942 [==============================] - 7s 7ms/step - loss: 5.4432 -
accuracy: 0.5513
Epoch 4/10
942/942 [==============================] - 7s 7ms/step - loss: 0.7846 -
accuracy: 0.5609
Epoch 5/10
942/942 [==============================] - 7s 7ms/step - loss: 0.6689 -
accuracy: 0.5621
Epoch 6/10
942/942 [==============================] - 7s 7ms/step - loss: 0.9381 -
accuracy: 0.5523
Epoch 7/10
942/942 [==============================] - 7s 7ms/step - loss: 0.6708 -
accuracy: 0.5552
Epoch 8/10
942/942 [==============================] - 7s 7ms/step - loss: 0.6884 -
accuracy: 0.5485
Epoch 9/10
942/942 [==============================] - 7s 7ms/step - loss: 0.7081 -
accuracy: 0.5442
Epoch 10/10
942/942 [==============================] - 7s 7ms/step - loss: 860.8011 -
accuracy: 0.5348
Epoch 1/10
942/942 [==============================] - 7s 7ms/step - loss: 0.6992 -
accuracy: 0.5070
Epoch 2/10
942/942 [==============================] - 7s 7ms/step - loss: 6.0464 -
accuracy: 0.5299
Epoch 3/10
942/942 [==============================] - 7s 7ms/step - loss: 0.6862 -
accuracy: 0.5486
Epoch 4/10
942/942 [==============================] - 7s 7ms/step - loss: 0.7231 -
accuracy: 0.5229
Epoch 5/10
942/942 [==============================] - 7s 7ms/step - loss: 0.6833 -
accuracy: 0.5285
Epoch 6/10
942/942 [==============================] - 7s 7ms/step - loss: 0.6846 -
accuracy: 0.5142
Epoch 7/10
```

```
942/942 [==============================] - 7s 7ms/step - loss: 9.9309 -
accuracy: 0.5345
Epoch 8/10
942/942 [==============================] - 7s 7ms/step - loss: 0.6588 -
accuracy: 0.5693
Epoch 9/10
942/942 [==============================] - 7s 7ms/step - loss: 0.6652 -
accuracy: 0.5680
Epoch 10/10
942/942 [==============================] - 7s 7ms/step - loss: 0.6825 -
accuracy: 0.5490
Epoch 1/10
942/942 [==============================] - 7s 8ms/step - loss: 7.1293 -
accuracy: 0.5325
Epoch 2/10
942/942 [==============================] - 7s 7ms/step - loss: 7.1293 -
accuracy: 0.5325
Epoch 3/10
942/942 [==============================] - 7s 7ms/step - loss: 7.1293 -
accuracy: 0.5325
Epoch 4/10
942/942 [==============================] - 7s 8ms/step - loss: 7.1293 -
accuracy: 0.5325
Epoch 5/10
942/942 [==============================] - 7s 7ms/step - loss: 7.1293 -
accuracy: 0.5325
Epoch 6/10
942/942 [==============================] - 7s 7ms/step - loss: 7.1293 -
accuracy: 0.5325
Epoch 7/10
942/942 [==============================] - 7s 7ms/step - loss: 7.1293 -
accuracy: 0.5325
Epoch 8/10
942/942 [==============================] - 7s 7ms/step - loss: 7.1293 -
accuracy: 0.5325
Epoch 9/10
942/942 [==============================] - 7s 7ms/step - loss: 7.1293 -
accuracy: 0.5325
Epoch 10/10
942/942 [==============================] - 7s 7ms/step - loss: 7.1293 -
accuracy: 0.5325
Epoch 1/10
942/942 [==============================] - 7s 7ms/step - loss: 8.2134 -
accuracy: 0.4675
Epoch 2/10
942/942 [==============================] - 7s 7ms/step - loss: 8.2134 -
accuracy: 0.4675
Epoch 3/10
```

```
942/942 [==============================] - 7s 7ms/step - loss: 8.2135 -
accuracy: 0.4675
Epoch 4/10
942/942 [==============================] - 7s 7ms/step - loss: 8.2134 -
accuracy: 0.4675
Epoch 5/10
942/942 [==============================] - 7s 7ms/step - loss: 8.2134 -
accuracy: 0.4675
Epoch 6/10
942/942 [==============================] - 7s 7ms/step - loss: 8.2135 -
accuracy: 0.4675
Epoch 7/10
942/942 [==============================] - 7s 7ms/step - loss: 8.2134 -
accuracy: 0.4675
Epoch 8/10
942/942 [==============================] - 7s 7ms/step - loss: 8.2134 -
accuracy: 0.4675
Epoch 9/10
942/942 [==============================] - 7s 7ms/step - loss: 8.2134 -
accuracy: 0.4675
Epoch 10/10
942/942 [==============================] - 7s 7ms/step - loss: 8.2135 -
accuracy: 0.4675
Epoch 1/10
942/942 [==============================] - 8s 7ms/step - loss: 4086976.7500 -
accuracy: 0.4895
Epoch 2/10
942/942 [==============================] - 7s 7ms/step - loss: 2.7370 -
accuracy: 0.5031
Epoch 3/10
942/942 [==============================] - 7s 8ms/step - loss: 4405.6025 -
accuracy: 0.5089
Epoch 4/10
942/942 [==============================] - 7s 8ms/step - loss: 0.4949 -
accuracy: 0.5246
Epoch 5/10
942/942 [==============================] - 8s 8ms/step - loss: 0.3867 -
accuracy: 0.5289
Epoch 6/10
942/942 [==============================] - 7s 7ms/step - loss: 0.7023 -
accuracy: 0.5388
Epoch 7/10
942/942 [==============================] - 7s 7ms/step - loss: 146.1113 -
accuracy: 0.5167
Epoch 8/10
942/942 [==============================] - 7s 7ms/step - loss: 9.4971 -
accuracy: 0.5122
Epoch 9/10
```

```
942/942 [==============================] - 7s 7ms/step - loss: 51.1580 -
accuracy: 0.5258
Epoch 10/10
942/942 [==============================] - 7s 7ms/step - loss: 9.4428 -
accuracy: 0.5205
Epoch 1/10
942/942 [==============================] - 7s 7ms/step - loss: 1141924.2500 -
accuracy: 0.4877
Epoch 2/10
942/942 [==============================] - 7s 7ms/step - loss: 1.0224 -
accuracy: 0.5009
Epoch 3/10
942/942 [==============================] - 7s 7ms/step - loss: 0.4764 -
accuracy: 0.5267
Epoch 4/10
942/942 [==============================] - 7s 7ms/step - loss: 0.3934 -
accuracy: 0.5433
Epoch 5/10
942/942 [==============================] - 7s 7ms/step - loss: 0.3932 -
accuracy: 0.5491
Epoch 6/10
942/942 [==============================] - 7s 7ms/step - loss: 0.6424 -
accuracy: 0.5500
Epoch 7/10
942/942 [==============================] - 7s 7ms/step - loss: 440.9635 -
accuracy: 0.5205
Epoch 8/10
942/942 [==============================] - 6s 7ms/step - loss: 126.8106 -
accuracy: 0.5206
Epoch 9/10
942/942 [==============================] - 6s 7ms/step - loss: 70.5557 -
accuracy: 0.5078
Epoch 10/10
942/942 [==============================] - 7s 7ms/step - loss: 42.1252 -
accuracy: 0.5332
```

[26]:
```
for model in models:
    model.evaluate(X_test, y_test)
```

```
237/237 [==============================] - 1s 2ms/step - loss: 0.7578 -
accuracy: 0.6200
237/237 [==============================] - 0s 2ms/step - loss: 0.6503 -
accuracy: 0.5354
237/237 [==============================] - 0s 1ms/step - loss: 7.0004 -
accuracy: 0.5409
237/237 [==============================] - 0s 1ms/step - loss: 8.3439 -
accuracy: 0.4591
237/237 [==============================] - 0s 1ms/step - loss: 3.1383 -
```

```
accuracy: 0.4679
237/237 [==============================] - 0s 2ms/step - loss: 0.9700 -
accuracy: 0.5503
```

1.3.1 Analysis

Model	Loss function	Accuracy
1	BinaryCrossentropy(from_logits=True)	0.6200
2	BinaryCrossentropy(from_logits=True)	0.5354
3	BinaryCrossentropy(from_logits=False)	0.5409
4	BinaryCrossentropy(from_logits=False)	0.4591
5	MeanSquaredError()	0.4679
6	MeanSquaredError()	0.5503

Model 1 seems to perform slightly above 50%, but this could be an outlier. Different models should be experimented with to determine if the animal's name, breed, color, age, sex, and other characteristics can predict whether or not it will be adopted.