To get all of this to run correctly we need to be in the correct python environment. Using Anaconda Here are the steps:

- conda create -n tf tensorflow
- conda activate tf
- conda install pandas
- conda install matplotlib

In [1]:

```
import pandas as pd
import numpy as np
import xml.etree.ElementTree as et
import matplotlib.pyplot as plt
%matplotlib inline
```

In [2]:

```
xml_file = 'stackexchange_data/diy.stackexchange.com/Posts_original.xml'
originaldf = pd.read_xml(xml_file,attrs_only=True,parser='etree')
originaldf.describe()
```

Out[2]:

	AcceptedAnswerId	AnswerCount	CommentCount	FavoriteCount	Id	LastE
count	22593.000000	64503.000000	173341.000000	7136.000000	173341.000000	60
mean	108373.832957	1.677674	1.950046	1.478840	118908.775829	34
std	70620.506794	1.453162	2.619226	2.210341	67767.548143	35
min	9.000000	0.000000	0.000000	0.000000	1.000000	
25%	41791.000000	1.000000	0.000000	1.000000	62355.000000	2
50%	106801.000000	1.000000	1.000000	1.000000	121874.000000	27
75%	170870.000000	2.000000	3.000000	1.000000	177914.000000	55
max	234205.000000	77.000000	48.000000	74.000000	234210.000000	141

In [3]:

```
originaldf.describe(exclude=[np.number])
```

Out[3]:

	Body	ContentLicense	CreationDate	LastActivityDate	L
count	173169	173341	173341	173341	
unique	173154	3	172934	137337	
top	There's no need to use this tag. When asking ...	CC BY-SA 3.0	2011-10-16T21:46:14.993	2010-07-21T19:33:18.130	2020-06-16T
freq	3	93808	2	2	

according to survey characteristics of good answers are :

- More varied vocabulary
- Answers referenced by other answers
- More comments from other users
- Earlier posted answers are likely to be better
- Answer most different from the rest
- Answer length (best)
- Forum specific easiest to look at are the answer length, time of posting and number of comments from other users. goal of this research is to find best answer. More interesting features are answers that are different from the rest. How to calculate answer similarity remains to be seen.. ###### start with comment count, answer length and time of posting? easy low hanging fruit

In [4]:
```
originaldf.loc[originaldf['PostTypeId'] == 2]
```

Out[4]:

	AcceptedAnswerId	AnswerCount	Body	CommentCount	ContentLicense	
7	NaN	NaN	<p>I've found that it works OK, but it's more ...	1	CC BY-SA 2.5	20
10	NaN	NaN	<p>I have used it for patching areas, but not ...	0	CC BY-SA 2.5	20
11	NaN	NaN	<p>I <em>just</em> caulked my shower last nigh...	3	CC BY-SA 2.5	2(
12	NaN	NaN	<p>It's just an ornamental wall it sounds like...	3	CC BY-SA 2.5	2(
13	NaN	NaN	<p>I just bought a permanent silicone product ...	3	CC BY-SA 2.5	20
...	...	...	...	...	...	
173330	NaN	NaN	<p>You have two choices (see sketch) - 1) Cont...	0	CC BY-SA 4.0	20:
173332	NaN	NaN	<p>i just had problems with my trimmer not goi...	0	CC BY-SA 4.0	20

	AcceptedAnswerId	AnswerCount	Body	CommentCount	ContentLicense	
173333	NaN	NaN	<p>A routine repair of 2 cycle engines is repl...	0	CC BY-SA 4.0	20
173335	NaN	NaN	<p>I take it by the fact that you linked to an...	0	CC BY-SA 4.0	20
173340	NaN	NaN	<p>To keep other gray water from	0	CC BY-SA 4.0	20

In [5]:
```
#html tags in body columns with blank space
originaldf.Body = originaldf.Body.str.replace('<[^>]*>','', regex=True)
```

In [6]:
```
# Need a difference between answer posting time and question posting time

from datetime import datetime

datestrings = originaldf.CreationDate.str.slice_replace(start=-4)

dateObjects = []
for i in range(len(datestrings)):
    dateObjects.append(datetime.strptime(datestrings[i],'%Y-%m-%dT%H:%M:%S'))

originaldf.CreationDate = dateObjects
```

In [7]:
```
# for x in originaldf.columns:
#     print(x, originaldf[x].dtypes)
```

In [8]:
```
# want the question posting time for each answer
# so merge each answer with its question along with the body and creation dat
df = pd.merge(left=originaldf.loc[originaldf['PostTypeId'] == 2,['Id', 'Creat
```

In [9]:
```
df['is_accepted_answer'] = df.Id_answer == df.AcceptedAnswerId
```

In [10]:
```
df
```

Out[10]:

	Id_answer	CreationDate_answer	Body_answer	CommentCount	ParentId	Id_question
0	9	2010-07-21 19:19:02	I've found that it works OK, but it's more dif...	1	3.0	3
1	12	2010-07-21 19:20:53	I have used it for patching areas, but not for...	0	3.0	3

	Id_answer	CreationDate_answer	Body_answer	CommentCount	ParentId	Id_question
2	13	2010-07-21 19:21:15	I just caulked my shower last night. I used GE...	3	2.0	2
3	14	2010-07-21 19:21:41	It's just an ornamental wall it sounds like, s...	3	1.0	1
4	15	2010-07-21 19:22:00	I just bought a permanent silicone product by ...	3	2.0	2
...	...	...	...	...	...	...
108210	234200	2021-09-04 23:57:43	You have two choices (see sketch) - 1) Continu...	0	146674.0	146674
108211	234202	2021-09-05 00:12:14	i just had problems with my trimmer not going ...	0	46938.0	46938
108212	234203	2021-09-05 00:25:11	A routine repair of 2 cycle engines is replace...	0	46938.0	46938
108213	234205	2021-09-05 01:19:37	I take it by the fact that you linked to an eB...	0	234187.0	234187
108214	234210	2021-09-05 04:35:47	To keep other gray water from backing up into ...	0	214731.0	214731

In [11]:

```
# Don't think i need this anymore

# need a variable to show that the post was selected or not: do a left join w
# left join variable is Id and right join variable is AcceptedAnswerId
# df = pd.merge(left=originaldf.loc[originaldf['PostTypeId'] == 2,['Id', 'Cre



# originaldata.loc[originaldata['column_name'] == some_value, [col_name1, col
```

In [12]:

```
df['time_difference'] = df.CreationDate_answer - df.CreationDate_question
```

In [13]:
```
for i in range(len(df.time_difference)):
    if str( df.time_difference[i] ) == 'NaT':
        print(df.time_difference[i])
```

In [14]:
```
time_difference_in_seconds = []

for i in range(len(df.time_difference)):
        time_difference_in_seconds.append(df.time_difference[i].total_seconds

df.time_difference = time_difference_in_seconds
```

In [15]:
```
df.AcceptedAnswerId.isna().sum()
```

Out[15]:
```
62026
```

In [16]:
```
originaldf.isna().sum()
```

Out[16]:
```
AcceptedAnswerId         150748
AnswerCount              108838
Body                        172
CommentCount                  0
ContentLicense                0
CreationDate                  0
FavoriteCount            166205
Id                            0
LastActivityDate              0
LastEditDate             112115
LastEditorUserId         112498
OwnerUserId                1916
PostTypeId                    0
Score                         0
Tags                     108838
Title                    108838
ViewCount                108838
ParentId                  65126
OwnerDisplayName         170670
CommunityOwnedDate       172872
LastEditorDisplayName    172946
ClosedDate               170901
dtype: int64
```

In [17]:
```
originaldf.loc[originaldf["PostTypeId"] == 1].shape
```

Out[17]:
```
(64503, 22)
```

In [18]:
```
originaldf.loc[originaldf["PostTypeId"] == 1].isna().sum()
```

Out[18]:
```
AcceptedAnswerId         41910
AnswerCount                  0
Body                         0
```

```
CommentCount                 0
ContentLicense               0
CreationDate                 0
FavoriteCount            57367
Id                           0
LastActivityDate             0
LastEditDate             31375
LastEditorUserId         31502
OwnerUserId                656
PostTypeId                   0
Score                        0
Tags                         0
Title                        0
ViewCount                    0
ParentId                 64503
OwnerDisplayName         63386
CommunityOwnedDate       64475
LastEditorDisplayName    64372
ClosedDate               62063
```

So it looks like only ~12000 of the 64000 questions have chosen answers. As there won't be reliable examples of chosen answers for the remaining 42000 we will remove them from the training set.

In [19]:
```
df.shape
```

Out[19]:
```
(108215, 11)
```

In [20]:
```
#Assume that if there are no AcceptedAnswerId for the question then
df.dropna(subset=["AcceptedAnswerId"],inplace=True)
df.reset_index(drop=True, inplace=True)
```

In [21]:
```
df.shape
```

Out[21]:
```
(46189, 11)
```

In [22]:
```
df.AcceptedAnswerId
```

Out[22]:
```
0             9.0
1             9.0
2            13.0
3            38.0
4            13.0
           ...
46184    234205.0
46185    234192.0
46186    234205.0
46187    234172.0
46188    234205.0
Name: AcceptedAnswerId, Length: 46189, dtype: float64
```

In [23]:
```
df.drop(['ParentId'], axis=1, inplace=True)
```

In [24]:
```
answer_lengths = []
for body in df.Body_answer:
    answer_lengths.append(len(body.split()))
df['answer_length'] = answer_lengths
```

In [25]:
```
df
```

Out[25]:

	Id_answer	CreationDate_answer	Body_answer	CommentCount	Id_question	Accepted
0	9	2010-07-21 19:19:02	I've found that it works OK, but it's more dif...	1	3	
1	12	2010-07-21 19:20:53	I have used it for patching areas, but not for...	0	3	
2	13	2010-07-21 19:21:15	I just caulked my shower last night. I used GE...	3	2	
3	14	2010-07-21 19:21:41	It's just an ornamental wall it sounds like, s...	3	1	
4	15	2010-07-21 19:22:00	I just bought a permanent silicone product by ...	3	2	
...	...	...	...	...	...	
46184	234190	2021-09-04 20:51:25	What to look for:\nIt will have a firm attachm...	0	234187	
46185	234192	2021-09-04 21:33:36	With the tabs intact the receptacles both are ...	2	234149	
46186	234195	2021-09-04 22:42:39	Dremel tools are high quality and go up to 350...	0	234187	
46187	234197	2021-09-04 23:16:53	It looks like something home made, and possibl...	1	234158	
46188	234205	2021-09-05 01:19:37	I take it by the fact that you linked to an eB...	0	234187	

46189 rows × 11 columns

Training The Neural Network

Now that there is a basic preprocessing of the data, we can train a classification model on it. As it stands right now there is no association between different answers that are in the same thread. Each entry in the datframe is an answer which could be classified as the chosen answer or not. This means that multiple questions could be classified by the algorithm as the chosen answer. We will see if this behaviour has a drastic effect. It seems more complicated to have the classifier choose only one answer per thread, so we will see how this more simple model performs before we move on to a more complicated data structure.

In [26]:
```
import tensorflow as tf
from tensorflow import keras
from sklearn.pipeline import Pipeline
from tensorflow.keras.wrappers.scikit_learn import KerasClassifier
```

In [27]:
```
print(tf.__version__)
print(keras.__version__)
```

```
2.0.0
2.2.4-tf
```

In [28]:
```
from sklearn.model_selection import train_test_split
```

In [29]:
```
# do a train test split on the data
test_size = 0.2
train_full_size = 1-test_size
dev_size = test_size/train_full_size
# get the features discussed above
features = df[['CommentCount', 'time_difference', 'answer_length']]
labels = df.is_accepted_answer
```

In [30]:
```
# From: https://stackoverflow.com/questions/34842405/parameter-stratify-from-
X_train_full, X_test, y_train_full, y_test = train_test_split(features, label
```

```
/Users/chris/opt/anaconda3/envs/tf/lib/python3.7/site-packages/sklearn/utils/
__init__.py:806: DeprecationWarning: `np.int` is a deprecated alias for the b
uiltin `int`. To silence this warning, use `int` by itself. Doing this will n
ot modify any behavior and is safe. When replacing `np.int`, you may wish to
use e.g. `np.int64` or `np.int32` to specify the precision. If you wish to re
view your current use, check the release note link for additional informatio
n.
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/de
vdocs/release/1.20.0-notes.html#deprecations
  return floored.astype(np.int)
/Users/chris/opt/anaconda3/envs/tf/lib/python3.7/site-packages/sklearn/utils/
__init__.py:806: DeprecationWarning: `np.int` is a deprecated alias for the b
uiltin `int`. To silence this warning, use `int` by itself. Doing this will n
```

```
ot modify any behavior and is safe. When replacing `np.int`, you may wish to
use e.g. `np.int64` or `np.int32` to specify the precision. If you wish to re
view your current use, check the release note link for additional informatio
n.
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/de
vdocs/release/1.20.0-notes.html#deprecations
```

In [31]:

```
# also create a dev set
X_train, X_dev, y_train, y_dev = train_test_split(X_train_full, y_train_full,
```

```
/Users/chris/opt/anaconda3/envs/tf/lib/python3.7/site-packages/sklearn/utils/
__init__.py:806: DeprecationWarning: `np.int` is a deprecated alias for the b
uiltin `int`. To silence this warning, use `int` by itself. Doing this will n
ot modify any behavior and is safe. When replacing `np.int`, you may wish to
use e.g. `np.int64` or `np.int32` to specify the precision. If you wish to re
view your current use, check the release note link for additional informatio
n.
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/de
vdocs/release/1.20.0-notes.html#deprecations
  return floored.astype(np.int)
/Users/chris/opt/anaconda3/envs/tf/lib/python3.7/site-packages/sklearn/utils/
__init__.py:806: DeprecationWarning: `np.int` is a deprecated alias for the b
uiltin `int`. To silence this warning, use `int` by itself. Doing this will n
ot modify any behavior and is safe. When replacing `np.int`, you may wish to
use e.g. `np.int64` or `np.int32` to specify the precision. If you wish to re
view your current use, check the release note link for additional informatio
n.
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/de
vdocs/release/1.20.0-notes.html#deprecations
  return floored.astype(np.int)
```

In [32]:

```
[X_train.shape,X_dev.shape,X_test.shape]
```

Out[32]:

```
[(27713, 3), (9238, 3), (9238, 3)]
```

In [33]:

```
# model = keras.models.Sequential()
# model.add(keras.layers.Input(shape=[3]))
# model.add(keras.layers.Dense(8, activation="sigmoid"))
# model.add(keras.layers.Dense(8, activation="sigmoid"))
# model.add(keras.layers.Dense(1, activation="sigmoid"))
```

In [34]:

```
def model():
    model = keras.models.Sequential([
        keras.layers.Input(shape=[3]),
        keras.layers.Dense(8, activation='relu'),
        keras.layers.Dense(1, activation='sigmoid')
    ])
    model.compile(loss='binary_crossentropy', optimizer='sgd', metrics=['accu
    return model
```

In [35]:
```
early_stopping_cb = keras.callbacks.EarlyStopping(patience=2,
                                                  restore_best_weights=True,
                                                  monitor='accuracy')
```

In [36]:
```
pipeline = Pipeline(steps=[('neuralnet',
                            KerasClassifier(build_fn=model,
                                            epochs=10,
                                            validation_data=(X_dev.values, y
                                            callbacks=[early_stopping_cb]
                                           ))])
```

In [37]:
```
history = pipeline.fit(X_train.values, y_train.values)
```

```
2021-12-06 21:46:06.942054: I tensorflow/core/platform/cpu_feature_guard.cc:1
45] This TensorFlow binary is optimized with Intel(R) MKL-DNN to use the foll
owing CPU instructions in performance critical operations:  SSE4.1 SSE4.2 AVX
AVX2 FMA
To enable them in non-MKL-DNN operations, rebuild TensorFlow with the appropr
iate compiler flags.
2021-12-06 21:46:06.942405: I tensorflow/core/common_runtime/process_util.cc:
115] Creating new thread pool with default inter op setting: 4. Tune using in
ter_op_parallelism_threads for best performance.
Train on 27713 samples, validate on 9238 samples
Epoch 1/10
27713/27713 [==============================] - 8s 275us/sample - loss: 101076
113.3353 - accuracy: 0.5120 - val_loss: 0.6925 - val_accuracy: 0.5115
Epoch 2/10
27713/27713 [==============================] - 6s 212us/sample - loss: 0.6923
- accuracy: 0.5120 - val_loss: 0.6925 - val_accuracy: 0.5115
Epoch 3/10
27713/27713 [==============================] - 6s 209us/sample - loss: 0.6923
- accuracy: 0.5120 - val_loss: 0.6925 - val_accuracy: 0.5115
Epoch 4/10
27713/27713 [==============================] - 4s 146us/sample - loss: 0.6923
- accuracy: 0.5120 - val_loss: 0.6925 - val_accuracy: 0.5115
```

In [38]:
```
# pd.DataFrame(history.history).plot(figsize=(8, 5))
# plt.grid(True)
# plt.gca().set_ylim(0, 1) # set the vertical range to [0-1]
# plt.show()
```

In [39]:
```
pipeline['neuralnet']
```

Out[39]:
```
<tensorflow.python.keras.wrappers.scikit_learn.KerasClassifier at 0x7fd005681
dd0>
```

In [40]:
```
X_train.shape[1:]
```

Out[40]:
```
(3,)
```

In [41]:
```
y_pred_train = pipeline.predict(X_train.values)
```

In [42]:
```
y_pred_train
```

Out[42]:
```
array([[False],
       [False],
       [False],
       ...,
       [False],
       [False],
       [False]])
```

In [43]:
```
y_pred_train.sum()
```

Out[43]:
```
39
```

In [44]:
```
from sklearn.metrics import classification_report
```

In [45]:
```
print(classification_report(y_train, y_pred_train))
```

```
              precision    recall  f1-score   support

       False       0.51      1.00      0.68     14158
        True       0.90      0.00      0.01     13555

    accuracy                           0.51     27713
   macro avg       0.70      0.50      0.34     27713
weighted avg       0.70      0.51      0.35     27713
```

so far all predictions are that the answers are poor. Now need to normalize the values for each thread as there can be a wide range in the number of comments, number of words in given answers or amount of time between question and answer.

In [46]:
```
df.groupby(['time_difference']).max()
```

Out[46]:

	Id_answer	CreationDate_answer	Body_answer	CommentCount	Id_question
time_difference					
0.0	215628	2021-02-02 21:58:28	Yes, but a few points need to be observed to m...	12	215627
34.0	58756	2015-01-31 22:08:32	After some experimentation, I found that the u...	0	58755

	Id_answer	CreationDate_answer	Body_answer	CommentCount	Id_question
time_difference					
41.0	5210	2011-03-17 22:16:09	Indeed†, it's a malfunction of some air valve ...	0	5209
50.0	32829	2013-10-16 17:59:35	The symbol to the left is the IEEE (Institute ...	3	32828
54.0	90871	2016-05-19 20:19:10	Yes! the tool is called a caulk gun. Use the s...	9	90870
...	...	...	...	...	...
344735073.0	227810	2021-06-23 19:10:57	I know this is an old post, but I'll give my 2...	1	21
344808888.0	231091	2021-07-30 08:03:21	The cheapest way is to mix nail polish remover...	0	1483
345057466.0	228686	2021-07-07 21:15:35	Live in Eastern Washington State and we have j...	1	847
348014707.0	232361	2021-08-21 17:17:05	I have seen this when the inside was kept cold...	0	1171