In [1]:

```
import ktrain
from ktrain import text
import pandas as pd
import random
import numpy as np
import math
```

In [2]:

```
csv_file = '../../data/merged_ktrain_google_en.csv'
data = pd.read_csv(csv_file).values
print(len(data))
```

```
21589
```

In [3]:

```
epochs = 3
learning_rate = 5e-5
batch_size = 32
max_length = 21
max_words = 25000
```

In [4]:

```
def split_test_data(data, split=0.1, random_seed=42):
    np.random.seed(random_seed)
    np.random.shuffle(data)
    split_item = math.floor(split * len(data))
    print('split at: ', split_item)
    x_test, y_test = data[:split_item, 0], data[:split_item, 1:]
    x_train, y_train = data[split_item:, 0], data[split_item:, 1:]
    return x_train, y_train, x_test, y_test
```

In [5]:

```
x_train, y_train, x_val, y_val = split_test_data(data, split=0.05, random_seed=4
242)
print(len(x_train), len(y_train), len(x_val), len(y_val))
```

```
split at:  1079
20510 20510 1079 1079
```

In [6]:

```
MODEL = 'distilbert-base-uncased'
transformer = text.Transformer(MODEL, maxlen=max_length, class_names=['less', 'equal', 'more'])
train_data = transformer.preprocess_train(x_train, y_train)
val_data = transformer.preprocess_test(x_val, y_val)
```

```
preprocessing train...
language: en
train sequence lengths:
        mean : 9
        95percentile : 15
        99percentile : 18


Is Multi-Label? False
preprocessing test...
language: en
test sequence lengths:
        mean : 9
        95percentile : 15
        99percentile : 19
```

In [7]:

```
model = transformer.get_classifier()
```

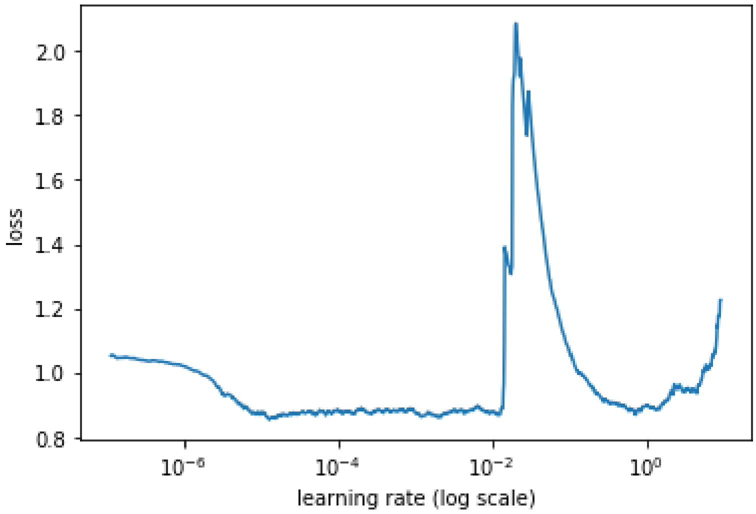
In [8]:

```
learner = ktrain.get_learner(model, train_data=train_data, val_data=val_data, batch_size=batch_size)
```

In [9]:

```
learner.lr_find(show_plot=True, max_epochs=2)
```

```
simulating training for different learning rates... this may take a
few moments...
Train for 640 steps
Epoch 1/2
640/640 [==============================] - 48s 74ms/step - loss: 0.9
229 - accuracy: 0.6628
Epoch 2/2
640/640 [==============================] - 42s 66ms/step - loss: 1.0
725 - accuracy: 0.6470


done.
Visually inspect loss plot and select learning rate associated with
falling loss
```



In [10]:

```
learner.fit_onecycle(5e-5, epochs=5)
```

```
begin training using onecycle policy with max lr of 5e-05...
Train for 641 steps, validate for 34 steps
Epoch 1/5
641/641 [==============================] - 50s 78ms/step - loss: 0.8
706 - accuracy: 0.6685 - val_loss: 0.8562 - val_accuracy: 0.6738
Epoch 2/5
641/641 [==============================] - 43s 67ms/step - loss: 0.8
564 - accuracy: 0.6690 - val_loss: 0.8423 - val_accuracy: 0.6766
Epoch 3/5
641/641 [==============================] - 44s 69ms/step - loss: 0.7
973 - accuracy: 0.6761 - val_loss: 0.8738 - val_accuracy: 0.6821
Epoch 4/5
641/641 [==============================] - 42s 65ms/step - loss: 0.5
316 - accuracy: 0.7919 - val_loss: 1.1134 - val_accuracy: 0.6172
Epoch 5/5
641/641 [==============================] - 43s 67ms/step - loss: 0.2
134 - accuracy: 0.9249 - val_loss: 1.5118 - val_accuracy: 0.6191
```

Out[10]:

```
<tensorflow.python.keras.callbacks.History at 0x7f1b9c2d7048>
```

In [11]:

```
learner.view_top_losses(n=10, preproc=transformer)
```

```
----------
id:783 | loss:7.45 | true:more | pred:equal)

----------
id:548 | loss:7.33 | true:more | pred:equal)

----------
id:375 | loss:7.3 | true:more | pred:equal)

----------
id:966 | loss:7.13 | true:more | pred:equal)

----------
id:907 | loss:7.08 | true:more | pred:equal)

----------
id:742 | loss:7.07 | true:more | pred:equal)

----------
id:143 | loss:7.04 | true:more | pred:equal)

----------
id:412 | loss:7.04 | true:more | pred:equal)

----------
id:0 | loss:7.02 | true:more | pred:equal)

----------
id:994 | loss:6.95 | true:more | pred:equal)
```

In [12]:

```
predictor = ktrain.get_predictor(learner.model, preproc=transformer)
```

In [13]:

```
predictor.explain(x_train[741])
```

Out[13]:

**y=equal** (probability **0.999**, score **6.778**) top features

| Contribution? | Feature |
|---:|---|
| +6.819 | Highlighted in text (sum) |
| -0.040 | <BIAS> |

european semester autumn package: creating an economy

In [14]:

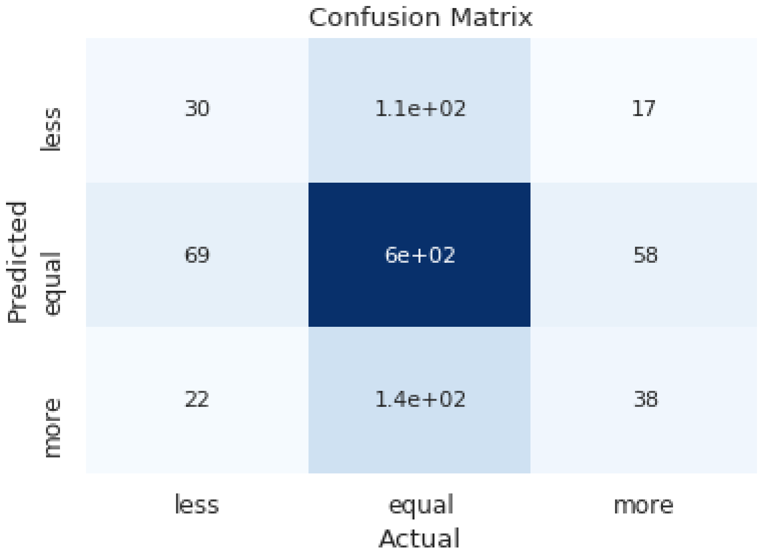
```
confusion = learner.evaluate()
```

```
              precision    recall  f1-score   support

           0       0.25      0.19      0.22       156
           1       0.71      0.83      0.76       727
           2       0.34      0.19      0.25       196

    accuracy                           0.62      1079
   macro avg       0.43      0.40      0.41      1079
weighted avg       0.58      0.62      0.59      1079
```

In [15]:

```
# print confusion matrix
import matplotlib.pyplot as plt
import seaborn as sn
labels = ['less', 'equal', 'more']
cm_df = pd.DataFrame(confusion, labels, labels)
sn.set(font_scale=1.1, font='Arial')
ax = sn.heatmap(cm_df, cmap="Blues", annot=True, annot_kws={"size": 11}, cbar=Fa
lse)
ax.set_xlabel("Actual")
ax.set_ylabel("Predicted")
ax.set_title("Confusion Matrix")
plt.show()
```

```
findfont: Font family ['Arial'] not found. Falling back to DejaVu Sa
ns.
findfont: Font family ['Arial'] not found. Falling back to DejaVu Sa
ns.
findfont: Font family ['Arial'] not found. Falling back to DejaVu Sa
ns.
```



In [ ]: