災難推文辨識

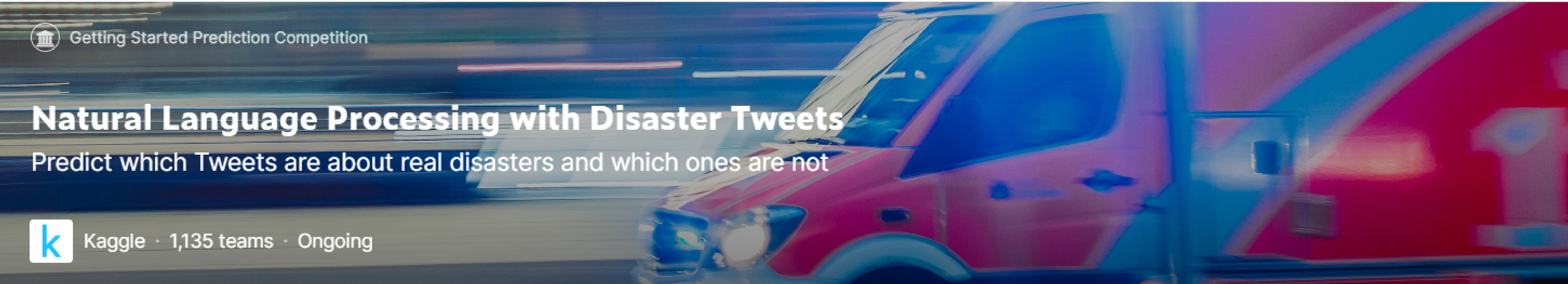
第十組

109590041 范遠皓
109590043 柯瑞霖

動機和目的

在Kaggle上，看到了這一個競賽，覺得有趣且實作出來是有應用價值的。

智能手機的普及使得人們能夠即時地發布他們觀察到的災難和緊急情況。因此，希望以編程方式來時時監測 Twitter 上的推文，讓救災組織可以在第一時間對發文的地點進行施救。



Getting Started Prediction Competition

Natural Language Processing with Disaster Tweets

Predict which Tweets are about real disasters and which ones are not

Kaggle · 1,135 teams · Ongoing

用到的技術

Google BERT

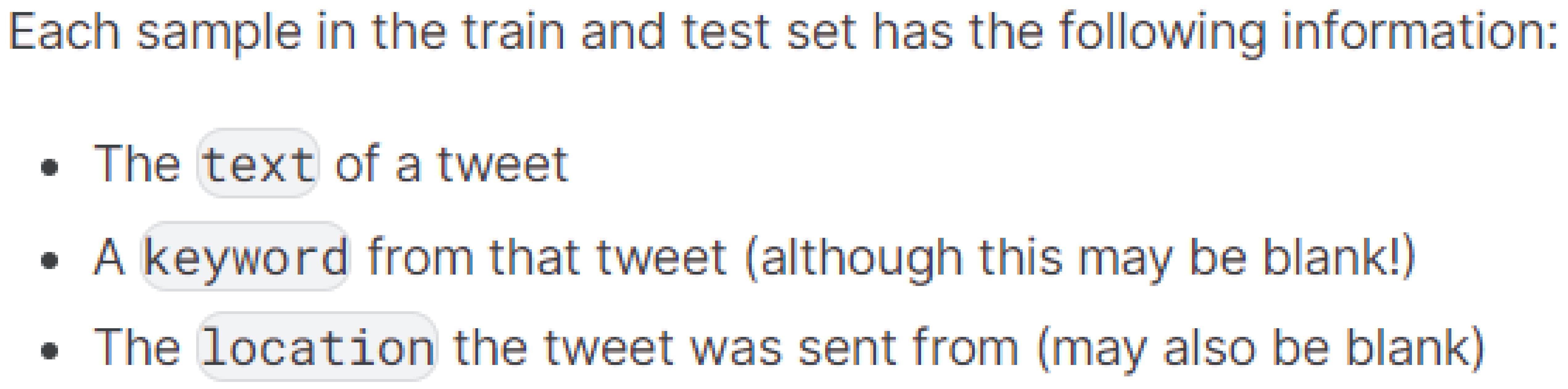
PyTorch

spaCy

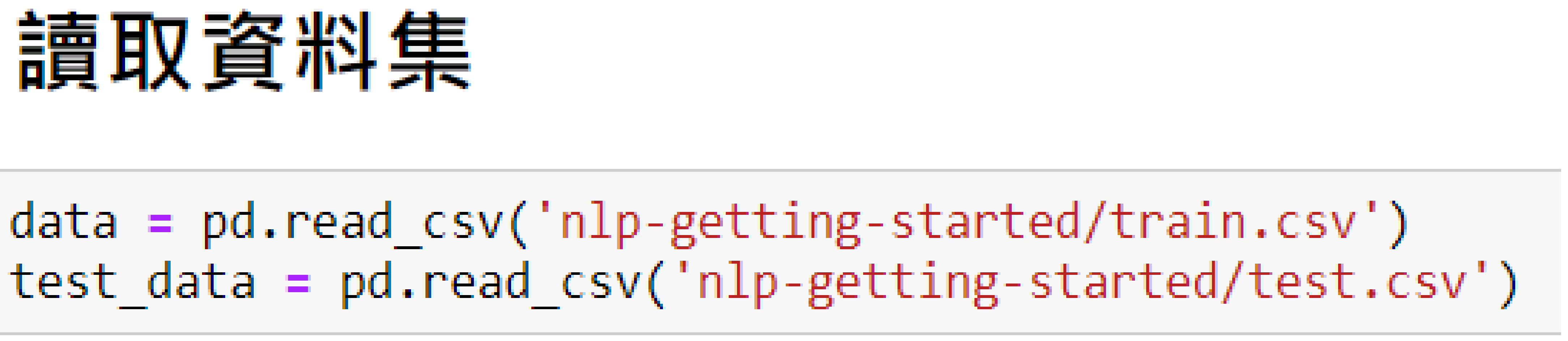
實作過程

- 讀取資料集
- 資料預處理
- 訓練模型
- 預測並輸出結果

資料集

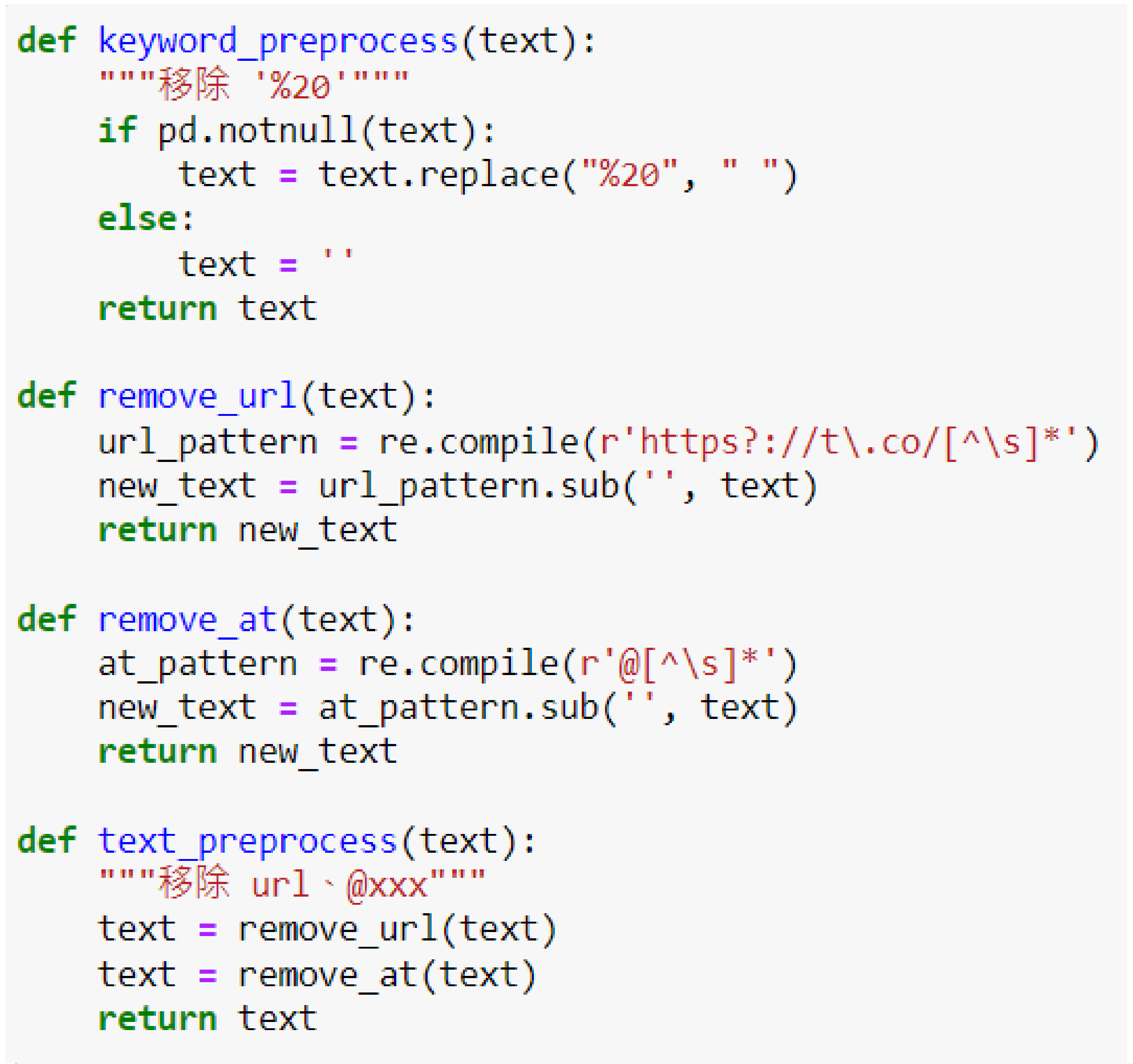

Each sample in the train and test set has the following information:

- The `text` of a tweet
- A `keyword` from that tweet (although this may be blank!)
- The `location` the tweet was sent from (may also be blank)


讀取資料集

```
data = pd.read_csv('nlp-getting-started/train.csv')
test_data = pd.read_csv('nlp-getting-started/test.csv')
```

資料預處理



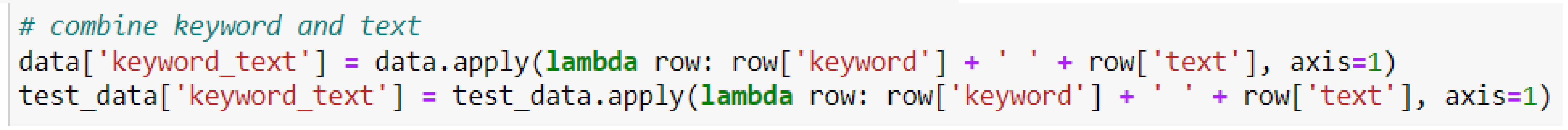
```python
def keyword_preprocess(text):
    """移除 '%20'"""
    if pd.notnull(text):
        text = text.replace("%20", " ")
    else:
        text = ''
    return text

def remove_url(text):
    url_pattern = re.compile(r'https?://t\.co/[^\s]*')
    new_text = url_pattern.sub('', text)
    return new_text

def remove_at(text):
    at_pattern = re.compile(r'@[^\s]*')
    new_text = at_pattern.sub('', text)
    return new_text

def text_preprocess(text):
    """移除 url、@xxx"""
    text = remove_url(text)
    text = remove_at(text)
    return text
```

將 keyword 標籤內有包含 '%20' 的替換掉

移除網址

移除@someone

將 keyword 加入 text(tweet)

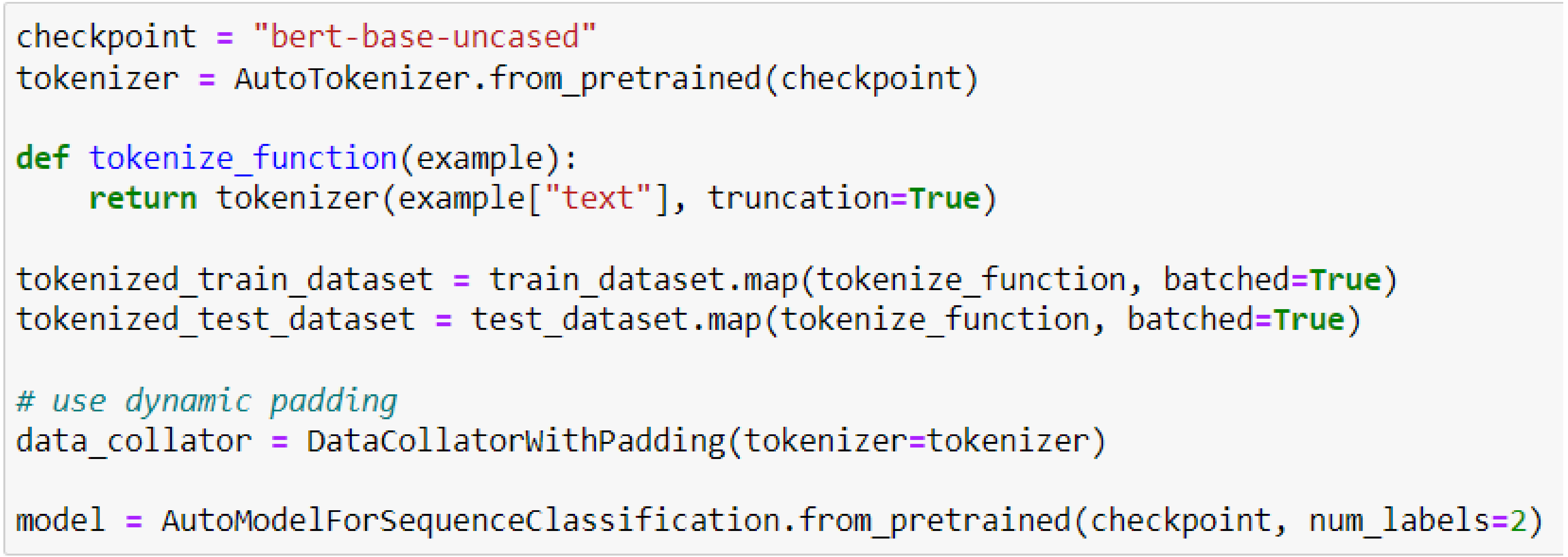


```python
# combine keyword and text
data['keyword_text'] = data.apply(lambda row: row['keyword'] + ' ' + row['text'], axis=1)
test_data['keyword_text'] = test_data.apply(lambda row: row['keyword'] + ' ' + row['text'], axis=1)
```

載入BERT模型

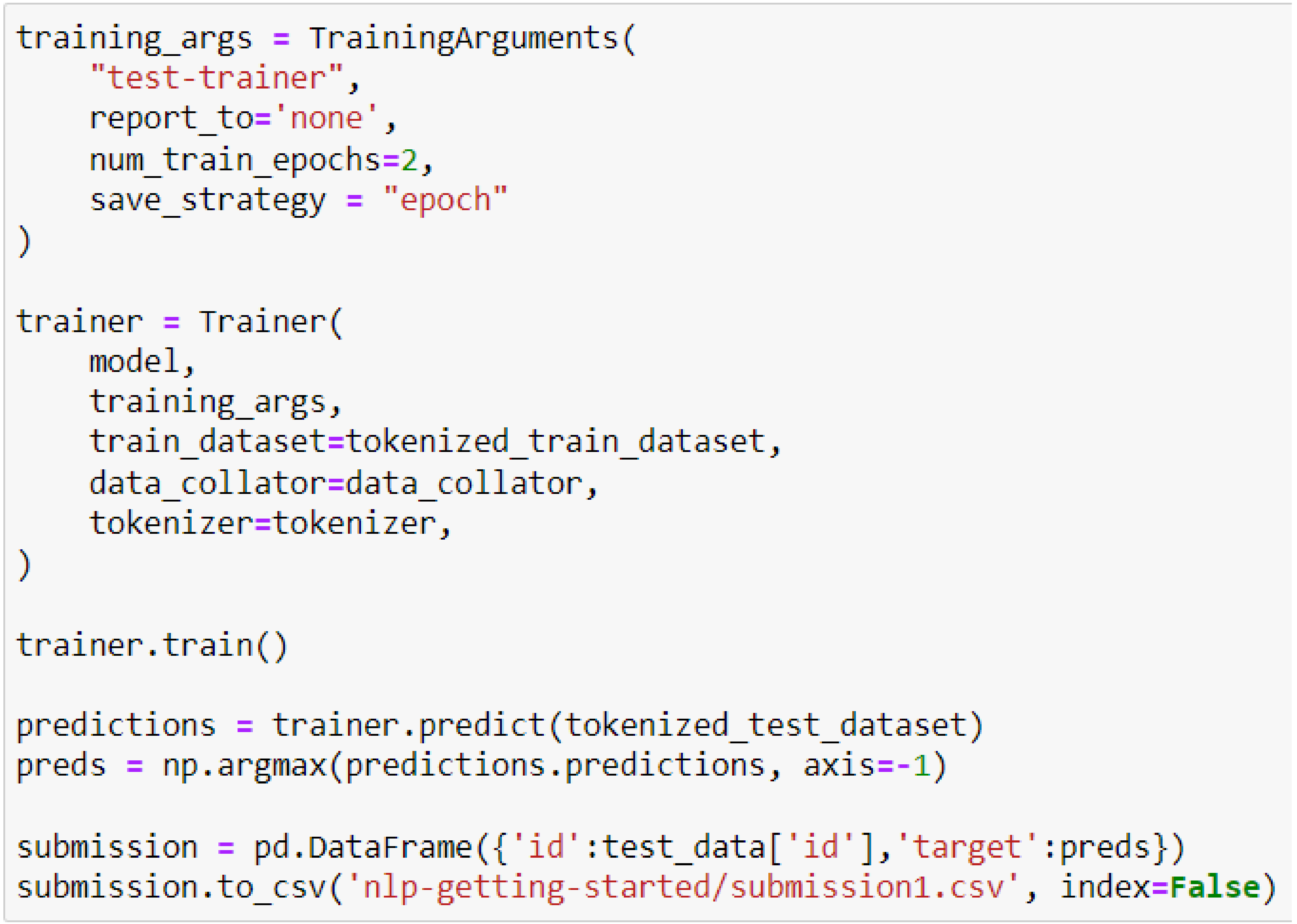
初始化一個 BERT tokenizer ，用於將文本轉換成 BERT 模型所需的輸入格式

載入預訓練模型BERT，用於訓練一個基於BERT模型的序列分類器



```python
checkpoint = "bert-base-uncased"
tokenizer = AutoTokenizer.from_pretrained(checkpoint)

def tokenize_function(example):
    return tokenizer(example["text"], truncation=True)

tokenized_train_dataset = train_dataset.map(tokenize_function, batched=True)
tokenized_test_dataset = test_dataset.map(tokenize_function, batched=True)

# use dynamic padding
data_collator = DataCollatorWithPadding(tokenizer=tokenizer)

model = AutoModelForSequenceClassification.from_pretrained(checkpoint, num_labels=2)
```

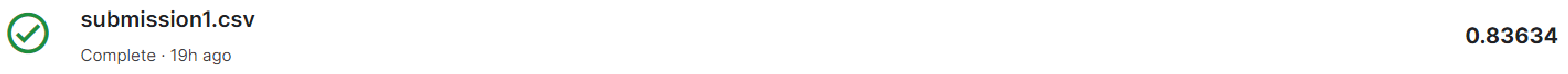
調整訓練參數並輸出結果



```python
training_args = TrainingArguments(
    "test-trainer",
    report_to='none',
    num_train_epochs=2,
    save_strategy = "epoch"
)

trainer = Trainer(
    model,
    training_args,
    train_dataset=tokenized_train_dataset,
    data_collator=data_collator,
    tokenizer=tokenizer,
)

trainer.train()

predictions = trainer.predict(tokenized_test_dataset)
preds = np.argmax(predictions.predictions, axis=-1)

submission = pd.DataFrame({'id':test_data['id'],'target':preds})
submission.to_csv('nlp-getting-started/submission1.csv', index=False)
```

提交結果

有預處理

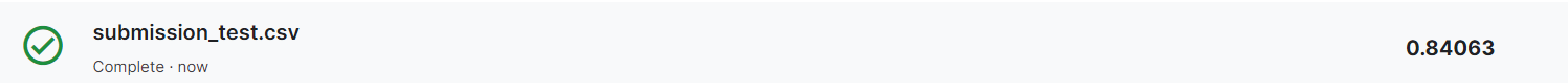


submission1.csv

Complete · 19h ago

0.83634

沒有預處理

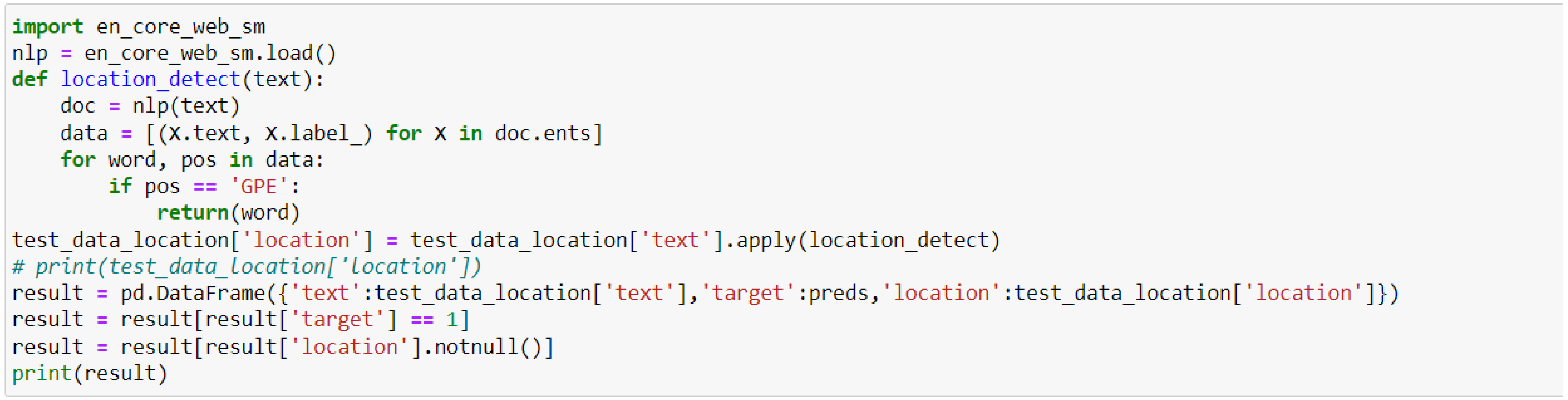


submission_test.csv

Complete · now

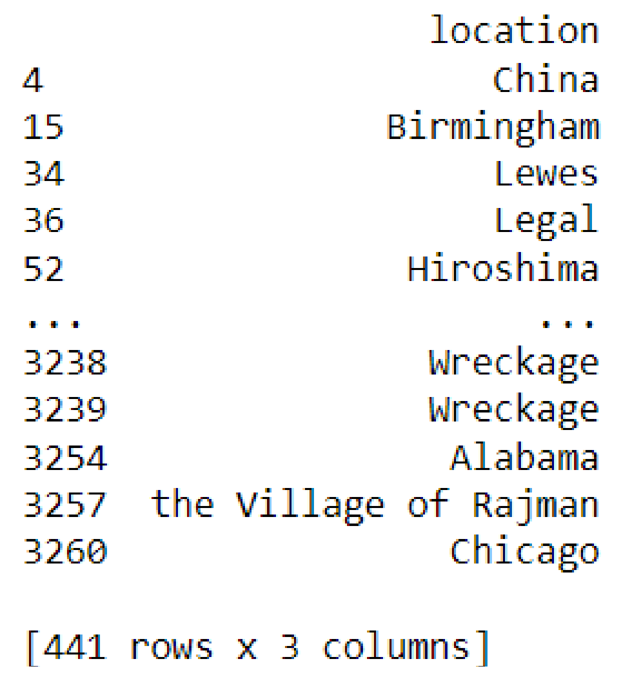
0.84063

找出受災地區



```python
import en_core_web_sm
nlp = en_core_web_sm.load()
def location_detect(text):
    doc = nlp(text)
    data = [(X.text, X.label_) for X in doc.ents]
    for word, pos in data:
        if pos == 'GPE':
            return(word)
test_data_location['location'] = test_data_location['text'].apply(location_detect)
# print(test_data_location['location'])
result = pd.DataFrame({'text':test_data_location['text'],'target':preds,'location':test_data_location['location']})
result = result[result['target'] == 1]
result = result[result['location'].notnull()]
print(result)
```

使用了spaCy套件的en_core_web_sm模型，檢測每條貼文中的地點



```
                   location
4                     China
15               Birmingham
34                    Lewes
36                    Legal
52                Hiroshima
...                     ...
3238               Wreckage
3239               Wreckage
3254                Alabama
3257  the Village of Rajman
3260                Chicago

[441 rows x 3 columns]
```

End