รายงาน

Data preprocessing (load_data function)

```python
def load_data(file_path):
    data = open(file_path).read()
    list_all = data.split('\n')
    sentence_in_list_all = []
    sentence = []
    answer_isus = []
    Y = []
    X = []
    # print(list_all)
    for i in list_all:
      if i == '':
        sentence_in_list_all.append(sentence)
        sentence = []
      else:
        sentence.append(i)
    for sentences in sentence_in_list_all:
      answer = []
      keep_all_word_and_ner_each_sentence = []
      for len_word in range(len(sentences)):
        word_and_ner = []
        word_and_ner_split = sentences[len_word].split("\t")
        word_and_ner.append(word_and_ner_split[1][2:])
        word_and_ner.append(word_and_ner_split[0])
        keep_all_word_and_ner_each_sentence.append(word_and_ner)
      # print(keep_all_word_and_ner_each_sentence)

      replaced = ['x' if x[0]=='' else x for x in keep_all_word_and_ner_each_sentence]
      # print(replaced)
      keep = []
      for len_word_and_ner in range(len(replaced)):
        if replaced[len_word_and_ner] != 'x':
          if len_word_and_ner == len(replaced)-1:
            list_answer = []
            # print("1: add",replaced[len_word_and_ner][1])
            keep.append(replaced[len_word_and_ner][1])
            list_answer.append(replaced[len_word_and_ner][0])
            list_answer.append(''.join(keep))
            answer.append(tuple(list_answer))
            # print('KEEP:',keep)
          elif replaced[len_word_and_ner][0] == replaced[len_word_and_ner+1][0]:
            # print("2: add",replaced[len_word_and_ner][1])
            keep.append(replaced[len_word_and_ner][1])
            # print('KEEP:',keep)
          elif replaced[len_word_and_ner][0] != replaced[len_word_and_ner+1][0]:
            list_answer = []
            # print("3: add",replaced[len_word_and_ner][1])
            keep.append(replaced[len_word_and_ner][1])
            # print('KEEP:',keep)
            list_answer.append(replaced[len_word_and_ner][0])
            list_answer.append(''.join(keep))
            answer.append(tuple(list_answer))
            keep = []
        else:
```

```
      # print(4)
      continue
  # print(answer)
  answer_isus.append(answer) #เอาไป evaluate
  list_tempX = []
  list_tempY = []
  for i in answer:
    list_tempX.append(i[1])
    list_tempY.append(i[0])
  X.append(list_tempX)
  # X.append(pos_tag(list_tempX, corpus='orchid_ud'))
  Y.append(list_tempY)
return (X,Y,answer_isus)
```

```
X_train, Y_train,evaluate_train = load_data('/content/drive/MyDrive/CONTEST 3/Contest 3 - NER/train_auto_tok.tsv')
X_dev, Y_dev,evaluate_dev = load_data('/content/drive/MyDrive/CONTEST 3/Contest 3 - NER/dev_auto_tok.tsv')
```

เริ่มจากเอา train_auto_tok.tsv มาเป็น data ของการเทรนโมเดลโดยที่ใช้ฟังก์ชัน load_data ทำกระบวนการภายใน

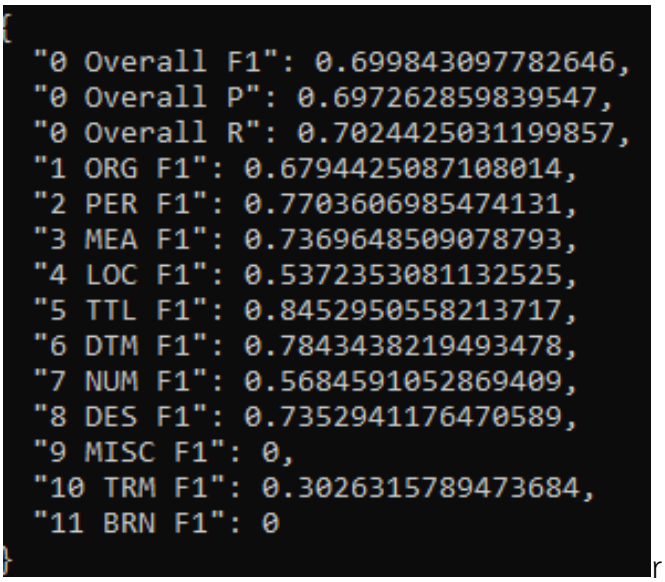
โดยใน load_data function จะทำการดึงตัวที่มี NER เหมือนกันมาติดกันเท่านั้น และผมได้ทำการใส่ part of speech โดยใช้ pythainlp จะได้หน้าตาของ X_train (1 sentence) เป็นรูปแบบประมาณนี้

```
1 X_train[-5]
```

```
[('เวลาประมาณ 09.20 น.', 'NOUN'),
 ('นาย', 'NOUN'),
 ('รองพล เจริญพันธ์', 'PROPN'),
 ('ทำเนียบรัฐบาล', 'NOUN'),
 ('รัฐมนตรี', 'NOUN'),
 ('สำนักนายกรัฐมนตรี', 'NOUN'),
 ('นาย', 'NOUN'),
 ('เนวิน ชิดชอบ', 'PROPN'),
 ('สำนักนายกรัฐมนตรี', 'PROPN'),
 ('คณะปฏิรูปฯ', 'NOUN'),
 ('กองบัญชาการ', 'NOUN'),
 ('กองทัพบก', 'NOUN')]
```

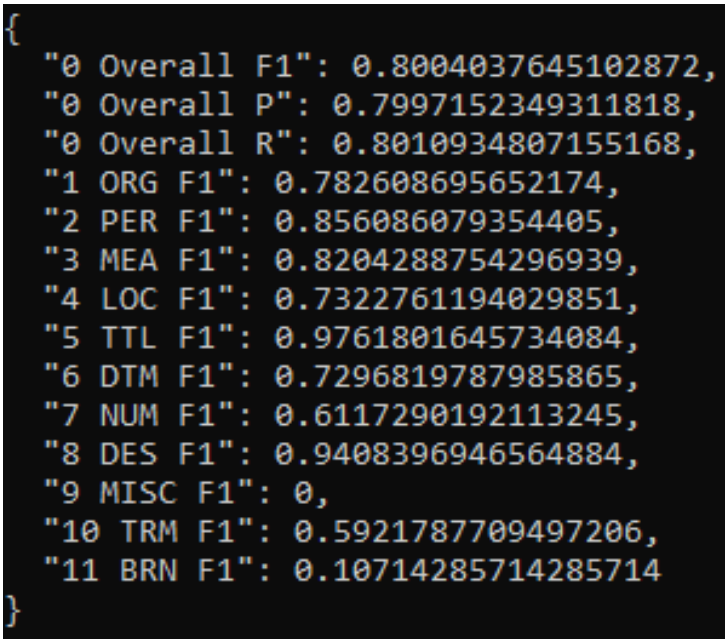
โดยเราได้เริ่มการทดลองโดยแบ่งตามfeatureที่จะนำมาใช้แล้วให้model predictแล้วevaluateกับ test ซึ่งเราจะได้overall F1-scoreมา

1. เริ่มจากการดึง data จาก train มาแล้วทำ CRF model แล้วจึงลอง evaluate กับ test

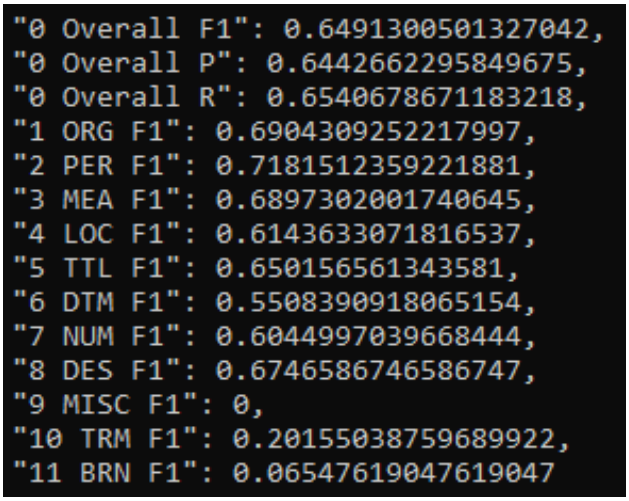


r

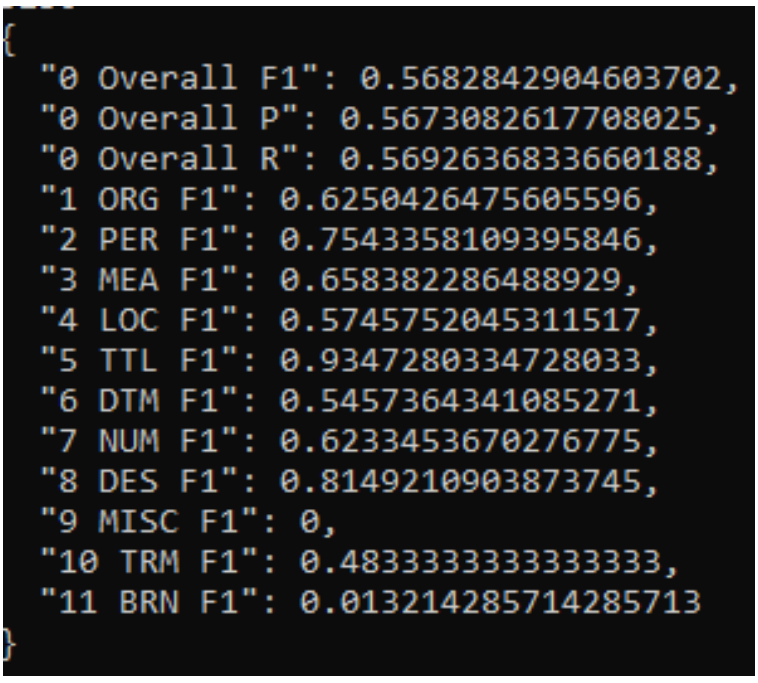
2. ทำfeature POS โดยใช้ library ของ pythainlp โดยTag words with corpus *orchid_udc* แล้วนำมา ทำCRF model แล้วจึงลองevaluateกับtest



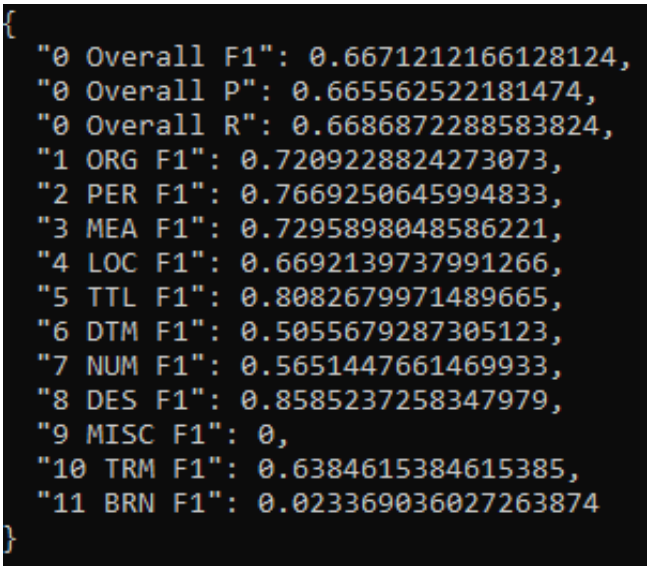
3.ทำfeature word window (bigram)+POSแล้วนำมาทำCRF model แล้วจึงลองevaluate กับ test



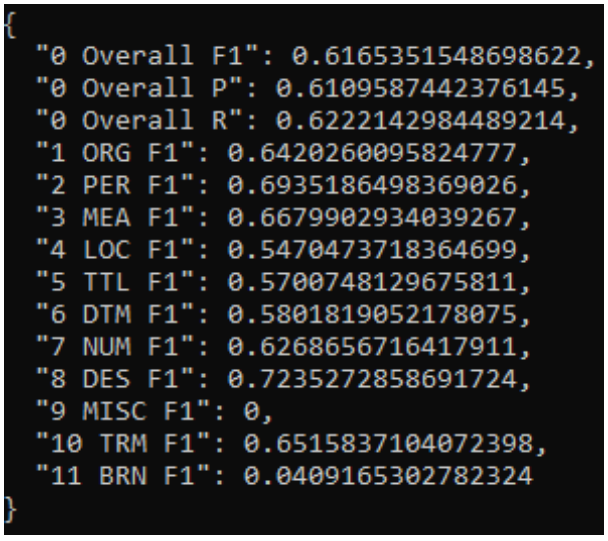
4.ทำpos tag window feature (bigram) +posแล้วนำมาทำCRF model แล้วจึงลองevaluate กับ test



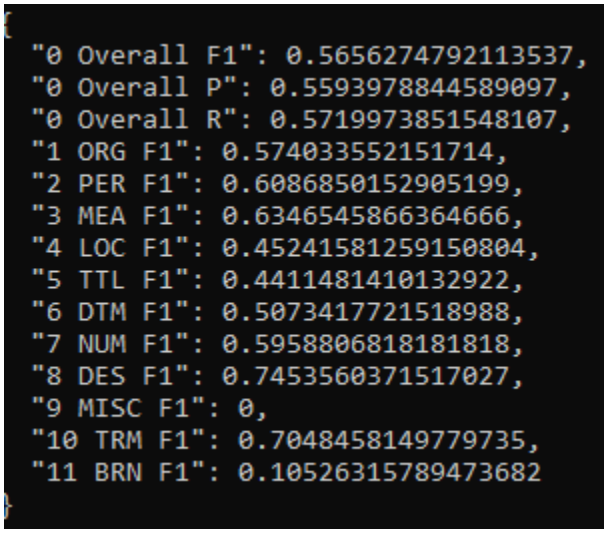
5.ทำpos conjunctive feature(bigram)+posแล้วนำมาทำCRF model แล้วจึงลองevaluate กับ test



6. feature word window (bigram)+ pos tag window(bigram) +posแล้วนำมาทำCRF model แล้วจึงลองevaluate กับtest



7.ทำการevaluateโดยใช้ featureที่ทำทั้งหมดแล้วนำมาทำCRF model แล้วจึงลองevaluate กับ test



จาก F1 scoreที่ลองทุกfeatureแล้วพบว่าทำfeature POS โดยใช้ library ของ pythainlp โดยTag words with corpus *orchid_ud*แล้วนำมาทำCRF model แล้วจึงลองevaluate กับtestจะได้F1 score สูงสุดซึ่งก็คือ 0.8