

IMDB 리뷰 분류

(긍정부정)

딥러닝 라이브러리, IMDB 데이터 불러오기



실행마다 동일한 결과를 얻기 위해 케라스에 랜덤 시드를 사용하고 텐서플로 연산을 결정적으로 만듭니다.

```
import tensorflow as tf
from tensorflow.keras.datasets import imdb

(train_input, train_target), (test_input, test_target) = imdb.load_data(
    num_words=200)
```

데이터 나누고, 패딩 인코딩 적용



```
from sklearn.model_selection import train_test_split
from tensorflow.keras.preprocessing.sequence import pad_sequences

train_input, val_input, train_target, val_target = train_test_split(
    train_input, train_target, test_size=0.2, random_state=42)

train_seq = pad_sequences(train_input, maxlen=100)
val_seq = pad_sequences(val_input, maxlen=100)
```

간단한 RNN 모델 설계 및 분류 모델



```
from tensorflow import keras
```

```
model = keras.Sequential()
```

```
model.add(keras.layers.SimpleRNN(8, input_shape=(100, 200)))
```

```
model.add(keras.layers.Dense(1, activation='sigmoid'))
```

원핫 인코딩 적용



```
train_oh = keras.utils.to_categorical(train_seq)
val_oh = keras.utils.to_categorical(val_seq)
```

순환신경망 학습하기




```
rmsprop = keras.optimizers.RMSprop(learning_rate=1e-4)
model.compile(optimizer=rmsprop, loss='binary_crossentropy',
              metrics=['accuracy'])

checkpoint_cb = keras.callbacks.ModelCheckpoint('best-simplernn-model.keras',
                                                save_best_only=True)
early_stopping_cb = keras.callbacks.EarlyStopping(patience=3,
                                                  restore_best_weights=True)


history = model.fit(train Oh, train_target, epochs=100, batch_size=64,
                    validation_data=(val Oh, val_target),
                    callbacks=[checkpoint_cb, early_stopping_cb])
```

학습에 따른 LOSS 확인하기



```
plt.plot(history.history['loss'])  
plt.plot(history.history['val_loss'])  
plt.xlabel('epoch')  
plt.ylabel('loss')  
plt.legend(['train', 'val'])  
plt.show()
```

단어 임베딩을 이용하여 텍스트 분류



```
model2 = keras.Sequential()  
  
model2.add(keras.layers.Embedding(200, 16, input_shape=(100,)))  
model2.add(keras.layers.SimpleRNN(8))  
model2.add(keras.layers.Dense(1, activation='sigmoid'))  
  
model2.summary()
```


단어 임베딩 적용한 모델 학습

```


rmsprop = keras.optimizers.RMSprop(learning_rate=1e-4)
model2.compile(optimizer=rmsprop, loss='binary_crossentropy',
               metrics=['accuracy'])

checkpoint_cb = keras.callbacks.ModelCheckpoint('best-embedding-model.keras',
                                                save_best_only=True)
early_stopping_cb = keras.callbacks.EarlyStopping(patience=3,
                                                  restore_best_weights=True)

history = model2.fit(train_seq, train_target, epochs=100, batch_size=64,
                    validation_data=(val_seq, val_target),
                    callbacks=[checkpoint_cb, early_stopping_cb])

```

학습에 따른 LOSS 확인

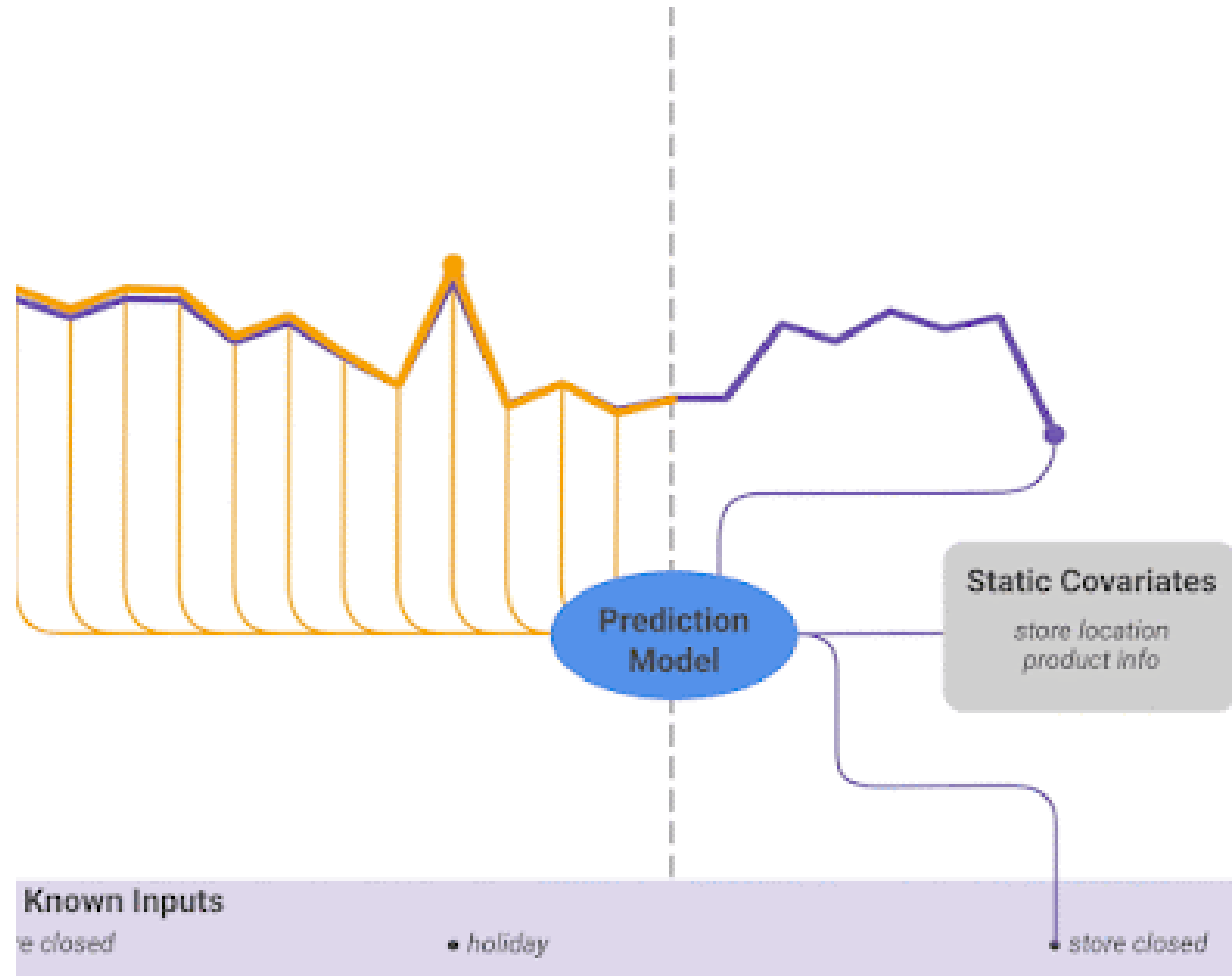


```
plt.plot(history.history['loss'])  
plt.plot(history.history['val_loss'])  
plt.xlabel('epoch')  
plt.ylabel('loss')  
plt.legend(['train', 'val'])  
plt.show()
```


실습 음악 악보 이어서 작곡

악보 dataset

How to?



라이브러리 import

- import numpy as np
- import tensorflow as tf

악보 코드 불러오기

- `txt = open('piano.txt','r').read()` # 학습할 데이터 불러오기

악보 코드 종류 구하기

- `bow = list(set(txt))` # 데이터 종류가 뭐가 있는지 추출
- `bow.sort()` # set이 순서가 없는 형태로 되어서 데이터가 정렬이 잘 안됨 -> 정렬해서 보기 쉽게
- `print(bow,len(bow))` # 데이터 종류, 데이터 종류 총 몇개 출력

데이터 전처리(코드 <-> 숫자)

- `txt_to_num = {}`
- `num_to_txt = {}`
- `preprocessTxt = []` # 전처리한 데이터 저장하기 위함
- `for i,j in enumerate(bow):` #
- `txt_to_num[j] = i` # `dict['A'] => 5` 변환 딕셔너니
- `num_to_txt[i] = j` # `dict[5] =? 'A'` 변환 딕셔너니

학습 데이터를 숫자로 변환

- # 텍스트로 구성된 txt를 숫자로 변환하는 과정
- preprocessTxt = []
- for i in txt:
- preprocessTxt.append(txt_to_num[i])

데이터 전처리 (RNN 학습할 수 있게)

- `train_x = []`
- `train_y = []`
- # 0~n 인덱스 속한 데이터를 활용하여 n+1 인덱스 예측 이걸 반복 (여기에선 n이 24)
- `for i in range(0, len(preprocessTxt)-25):`
 - `train_x.append(preprocessTxt[i:i+25])`
 - `train_y.append(preprocessTxt[i+25])`
- `train_x = np.array(train_x)`
- `train_y = np.array(train_y)`

원핫 인코딩

- `train_x = tf.one_hot(train_x, len(bow))`
- `train_y = tf.one_hot(train_y, len(bow))`

모델 구조 작성

```
model = tf.keras.models.Sequential([  
    tf.keras.layers.LSTM(256,input_shape=(train_x.shape[1:]),return_sequences=True),  
    tf.keras.layers.LSTM(128),  
    tf.keras.layers.Dense(len(bow),activation='softmax')  
])
```

모델 설정 & 학습

- `model.compile(loss='categorical_crossentropy',optimizer='adam')`
- `model.fit(train_x,train_y,batch_size=64,epochs=30,verbose=2)`

예측하기

```
# 예측한 이후의 인덱스 번호 -> 따로 텍스트 파일 저장하기 위함
예측한인덱스 = len(preprocessTxt)+1
for i in range(300):
    다음예상 = preprocessTxt[-25:]

    다음예상 = tf.one_hot(다음예상,31)

    다음예상 = tf.expand_dims(다음예상,axis=0)

    다음예상input = np.argmax(model.predict(다음예상)[0])

    preprocessTxt.append(다음예상input)

print(len(preprocessTxt))
```

숫자에서 악보 코드로 변환

- # 처음부터 예측한 내용까지 악보 만들어보기
- `total_music = []`
- `for i in preprocessTxt:`
- `total_music.append(num_to_txt[i])`
- `result = ''.join(total_music) print(result)`

악보코드 -> 피아노

[tradtunedb tune bank](#)