

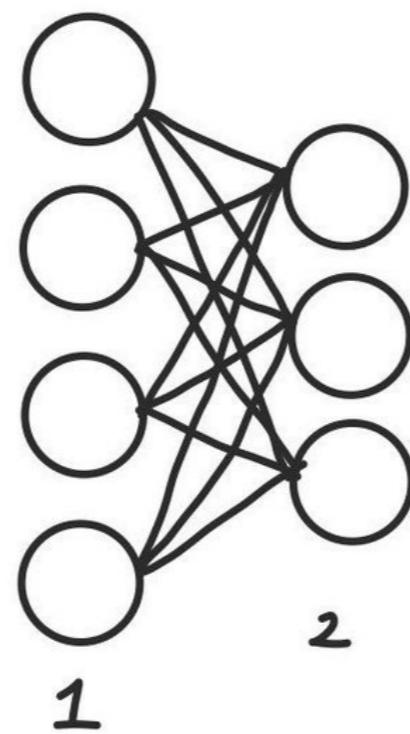
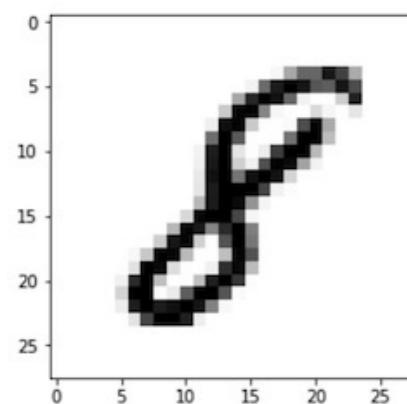
개요

- 간단한 딥러닝 프로그램 예제를 돌려보고, 이에 익숙해진다
- 이진 분류(Binary Classification) : 영화 리뷰 (좋다/나쁘다)
- 다중 분류(Multiclass classification) : 뉴스 기사 분류 (정치, 사회, 스포츠, 문화, 영화, ...)
- 회귀 분석 (Regression) : 미래 집값 예측

프로세스

- 데이터셋 (어떤 데이터셋인가?, 어떻게 구성되어 있나?)
- 데이터 전처리
- 모델 수립
- 학습/검증
- 시험/평가/가시화

예) MNIST 데이터셋 학습



데이터셋
+
전처리

모델 수립

Loss 함수 결정

Value

이게 얼마가 될거 같니?

output을
그냥 받는다.

MSE: Mean Squared Error

O/X

기냐? 아니냐?

output에
sigmoid를 먹인다.

Binary CrossEntropy

Category

종류중에 요건 뭐나?

output에
softmax를 먹인다.

Categorical CrossEntropy

뉴스 기사 토픽 분류

- 다중 분류(Multiclass classification) : 뉴스 기사 분류 (정치, 사회, 스포츠, 문화, 영화 등 총 46개 분류)
- 정답 코드를 참고하지 않고, 위의 라인과 이전 문제(영화 리뷰 긍부정)의 코드를 이용해 보자



World US Politics Business Health Entertainment Style Travel Sports Videos

CNN International - Breaking News, US News, World News ...

Find the latest breaking news and information on the top stories, weather, business, entertainment, politics, and more. For in-depth coverage, CNN provides ...

World

View CNN world news today for international news and videos ...

Latest News

View the latest news and headlines on CNN.com.

US Politics

Politics at CNN has news, opinion and analysis of American and ...

[More results from cnn.com »](#)

US

View the latest US news, top stories, photos and videos from ...

Breaking News, Latest News ...

Scenes from the field - Nepal's Organ Trail - Healthiest cities - ...

Latest News Videos

Catch up on the latest news videos from CNN.

데이터 셋

```
from keras.datasets import reuters  
  
(train_data, train_labels), (test_data, test_labels) = reuters.load_data(num_words=10000)
```

- Keras에서 기본 데이터 셋으로 제공
- num_words는 기사에 나타난 단어 중 가장 자주 나타나는 단어 10000개만 사용하겠다는 의미
- 핵심: 어떻게 각 리뷰 데이터를 고차원의 점 하나로 표현

전처리

- 각 리뷰 데이터는 10,000개의 단어만을 사용
- 하지만, 각 **뉴스**의 길이는 서로 다르다
- 모델에 입력하는 input의 길이는 일정하다. 해결책은?

One-Hot Encoding



color	color_red	color_blue	color_green
red	1	0	0
green	0	0	1
blue	0	1	0
red	1	0	0

One-Hot Encoding

```
import numpy as np

def vectorize_sequences(sequences, dimension=10000):
    # 크기가 (len(sequences), dimension)이고 모든 원소가 0인 행렬을 만듭니다
    results = np.zeros((len(sequences), dimension))
    for i, sequence in enumerate(sequences):
        results[i, sequence] = 1. # results[i]에서 특정 인덱스의 위치를 1로 만듭니다
    return results

# 훈련 데이터를 벡터로 변환합니다
x_train = vectorize_sequences(train_data)
# 테스트 데이터를 벡터로 변환합니다
x_test = vectorize_sequences(test_data)
```

- 차원의 개수만큼 공간을 할당하고 미리 0으로 초기화
- 뉴스의 각 단어에 해당하는 위치만 1로 세팅한다.

전처리 : One-Hot Encoding 후

```
# 훈련 데이터를 벡터로 변환합니다  
x_train = vectorize_sequences(train_data)  
# 테스트 데이터를 벡터로 변환합니다  
x_test = vectorize_sequences(test_data)
```

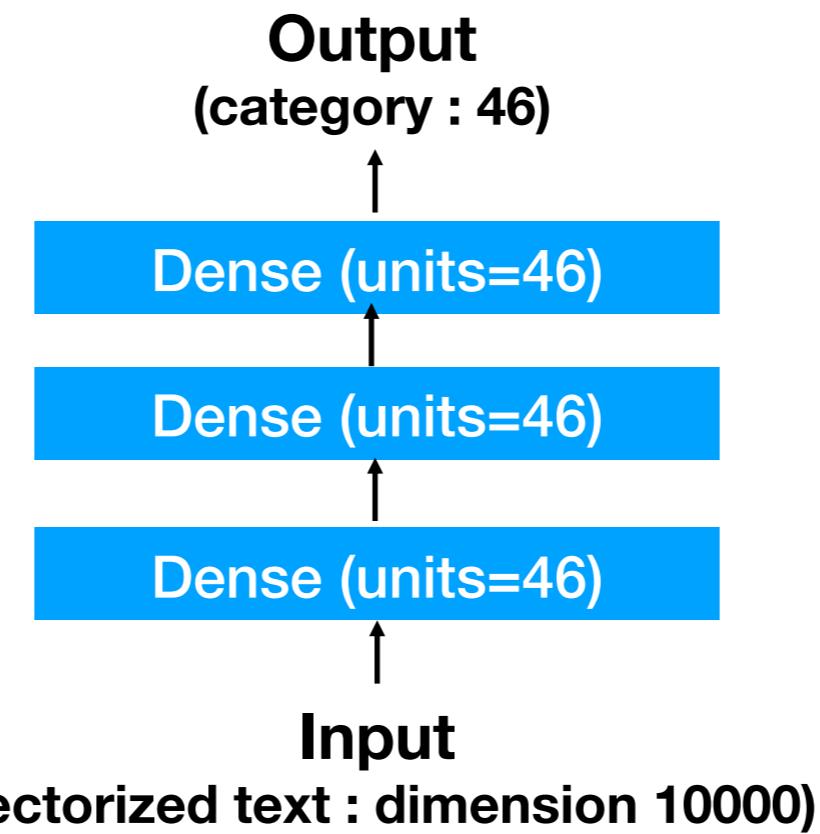
이제 샘플은 다음과 같이 나타납니다:

```
x_train[0]  
  
array([0., 1., 1., ..., 0., 0., 0.])
```

신경망 모델 만들기

- 모델을 코딩해 보자
- `m = models.Sequential()`
- `m.add(layers.Dense(...))`
- ...
- 입력 차원은 10000 차원의 데이터 (One-hot)
- 출력 차원은 1차원 (~~Positive일 확률~~) 46차원 (뉴스 분류 개수)

신경망 모델 만들기



```
from keras import models
from keras import layers

model = models.Sequential()
model.add(layers.Dense(64, activation='relu', input_shape=(10000,)))
model.add(layers.Dense(64, activation='relu'))
model.add(layers.Dense(46, activation='softmax'))
```

최적화

- Optimizer는?
- Loss 함수는?
- Metrics는?

최적화

- Optimizer는?
- Loss 함수는?
- Metrics는?

```
model.compile(optimizer='rmsprop',
              loss='categorical_crossentropy',
              metrics=[ 'accuracy' ])
```

학습

```
x_val = x_train[:10000]
partial_x_train = x_train[10000:]

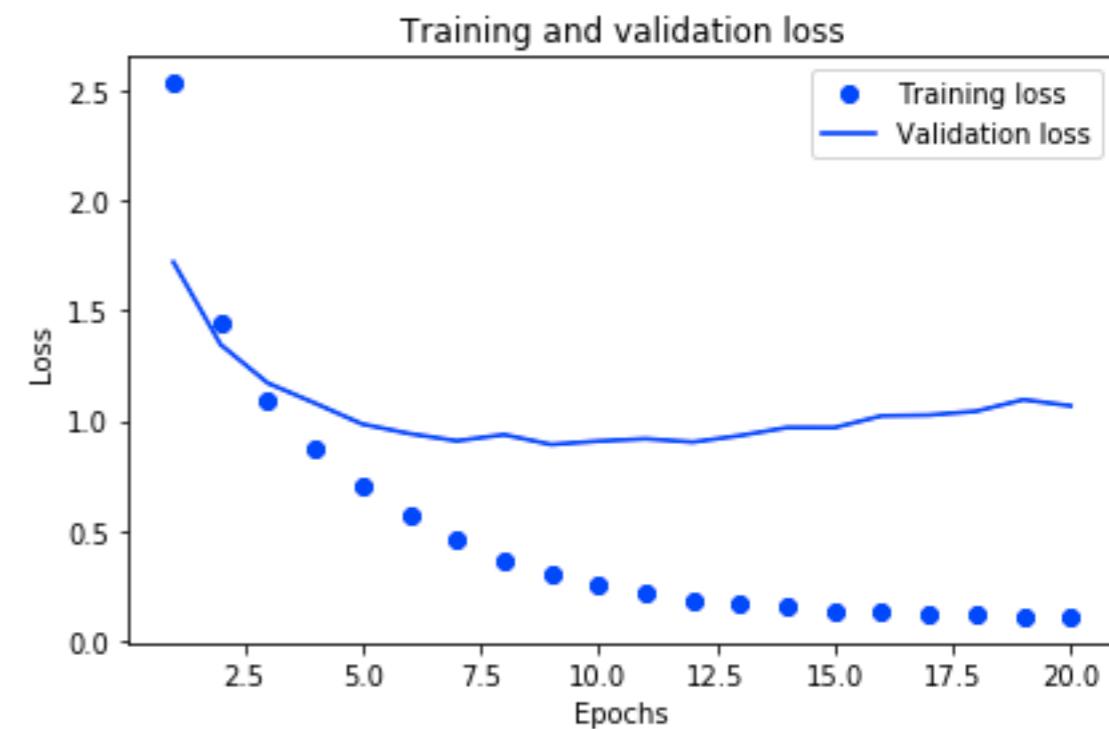
y_val = y_train[:10000]
partial_y_train = y_train[10000:]
```

```
history = model.fit(partial_x_train,
                     partial_y_train,
                     epochs=20,
                     batch_size=512,
                     validation_data=(x_val, y_val))
```

학습 결과 시각화

```
history_dict = history.history  
history_dict.keys()  
  
dict_keys(['val_loss', 'val_acc', 'loss', 'acc'])
```

```
import matplotlib.pyplot as plt  
  
acc = history.history['acc']  
val_acc = history.history['val_acc']  
loss = history.history['loss']  
val_loss = history.history['val_loss']  
  
epochs = range(1, len(acc) + 1)  
  
# 'bo'는 파란색 점을 의미합니다  
plt.plot(epochs, loss, 'bo', label='Training loss')  
# 'b'는 파란색 실선을 의미합니다  
plt.plot(epochs, val_loss, 'b', label='Validation loss')  
plt.title('Training and validation loss')  
plt.xlabel('Epochs')  
plt.ylabel('Loss')  
plt.legend()  
  
plt.show()
```



학습 결과 시각화

```
plt.clf()      # 그래프를 초기화합니다
acc = history_dict['acc']
val_acc = history_dict['val_acc']

plt.plot(epochs, acc, 'bo', label='Training acc')
plt.plot(epochs, val_acc, 'b', label='Validation acc')
plt.title('Training and validation accuracy')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend()

plt.show()
```

