영화 리뷰 감성 분석

1. 영화 리뷰 텍스트를 사용하여 감성을 분류하는 방법.
2. 단어 빈도(Bag of Words)를 사용하여 텍스트를 수치화.
3. 로지스틱 회귀와 소프트맥스 감성 분류기 구축.
4. 분류기의 정확도를 측정하고 ROC 곡선을 계산하여 분류기의 효과 평가.

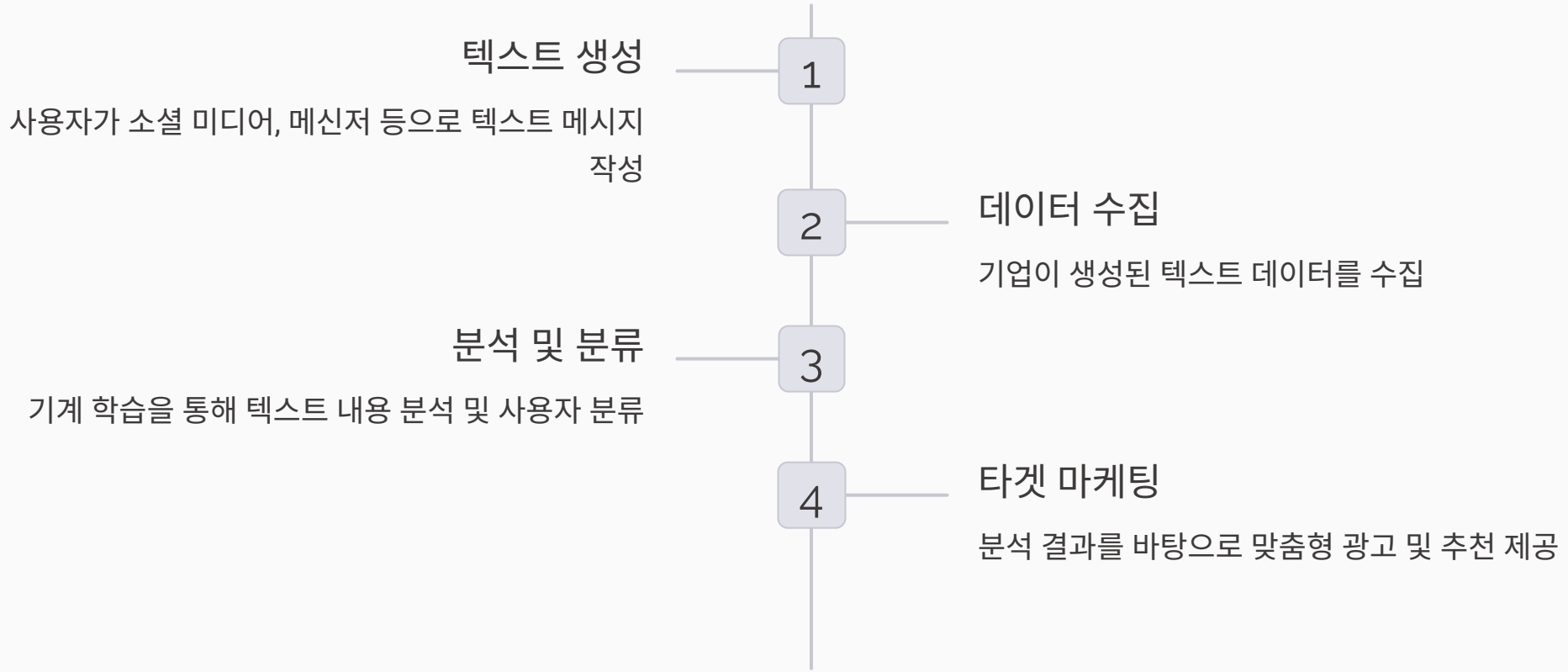
안재목 교수 - 한림대 소프트웨어학부

텍스트 데이터의 중요성

오늘날 소셜 미디어, SMS, 메신저 앱 등을 통해 매일 수천억 건의 텍스트 메시지가 생성.

소셜 미디어 기업, 통신사, 앱 개발자들은 사용자가 보내는 메시지를 분석하여 의사 결정을 내리고 사용자를 분류.

예를 들어, 점심으로 먹은 태국 음식에 대해 문자 메시지를 보낸 후 소셜 미디어에 태국 레스토랑 광고가 표시되는 경험은 빅데이터 분석의 한 예시.



감성 분석: 긍정 혹은 부정

영화 리뷰를 분석하여 제품에 대한 만족도를 파악. 리뷰는 긍정적이거나 부정적인 감정을 담고 있음.

감성을 분류할 수 있다면, 특정 영상이 시청자들에게 어떤 감정을 불러일으켰는지 파악할 수 있고, 더 나아가 그 감정이 사용자의 다음 행동(예: 관련 상품 구매)과 어떤 연관이 있는지 분석.

긍정적 리뷰

"와우, 정말 좋은 영화였어요! 빌의 연기가 훌륭했습니다!"

부정적 리뷰

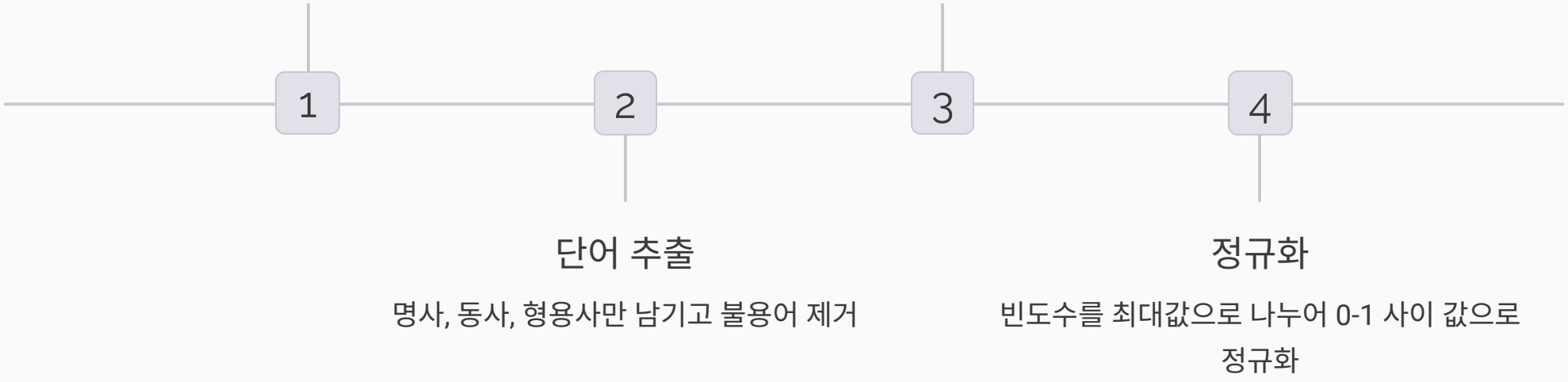
"이 영화는 너무 부적절했고, 3시간이 넘게 지루했으며, 폭력적인 장면에 불쾌감을 느꼈습니다."

감성 분석

기계 학습 모델이 리뷰 텍스트를 분석하여 긍정/부정 감성을 분류

Bag of Words 소개

Bag of Words 모델은 자연어 처리(NLP)에서 사용되는 방법으로, 문장 형태의 텍스트를 입력받아 특징 벡터로 변환. 각 단어의 빈도수 카운트가 마치 "가방"과 같아서 Bag of words(BoW)라고 부름.



영화 리뷰 - 입력 데이터

Index	id	sentiment	review
0	"5814_8"	1	"With all this stuff going down at the moment with MJ i've started listening to his music…
1	"2381_9"	1	"\"The Classic War of the Worlds\" by Timothy Hines is a very entertaining film that obvi…
2	"7759_3"	0	"The film starts with a manager (Nicholas Bell) giving welcome investors (Robert Carradin…
3	"3630_4"	0	"It must be assumed that those who praised this film (\"the greatest filmed opera ever,\"…
4	"9495_8"	1	"Superbly trashy and wondrously unpretentious 80's exploitation, hooray! The pre-credits …
5	"8196_8"	1	"I dont know why people think this is such a bad movie. Its got a pretty good plot, some …
6	"7166_2"	0	"This movie could have been very good, but comes up way short. Cheesy special effects and…
7	"10633_1"	0	"I watched this video at a friend's house. I'm glad I did not waste money buying this one…
8	"319_1"	0	"A friend of mine bought this film for £1, and even then it was grossly overpriced. Despi…
9	"8713_10"	1	" This movie is full of references. Like \"Mad Max II\", \"The wild one\" and …
10	"2486_3"	0	"What happens when an army of wetbacks, towelheads, and Godless Eastern European commies …
11	"6811_10"	1	"Although I generally do not like remakes believing that remakes are waste of time; this …
12	"11744_9"	1	"\"Mr. Harvey Lights a Candle\" is anchored by a brilliant performance by Timothy Spall.<…
13	"7369_1"	0	"I had a feeling that after \"Submerged\", this one wouldn't be any better... I was right…

리뷰 데이터 정제-코드

```
def review_to_words(raw_review):
    soup = BeautifulSoup("<html>" + raw_review + "</html>",'html.parser')
    review_text = soup.get_text()
    letters_only = re.sub("[^a-zA-Z]", " ", review_text)
    words = letters_only.lower().split()
    stops = set(stopwords.words("english"))
    meaningful_words = [w for w in words if not w in stops]
    return " ".join(meaningful_words)

x_train_origin = pd.read_csv("C:\\Users\\ajm\\Desktop\\MyDir\\labeledTrainData.tsv", header=0, delimiter="\t",
quoting=3)
y_train_origin = x_train_origin['sentiment'].values
x_train_short = x_train_origin[:6000]
y_train = y_train_origin[:6000]

clean_train_reviews = []
for i in range(0, x_train_short["review"].size):
    clean_train_reviews.append(review_to_words(x_train_short["review"][i]))
```

Bag of Words(BoW) -코드

텍스트 데이터를 숫자로 변환. CountVectorizer 이용.

1. CountVectorizer 객체 생성 (max_features=5000으로 설정)
2. fit_transform 메서드로 어휘 학습 및 벡터화

```
vectorizer = CountVectorizer(analyzer='word',max_features = 5000)
x_train = vectorizer.fit_transform(clean_train_reviews).toarray()
x_train, x_test, y_train, y_test = train_test_split(x_train, y_train, test_size=0.3, random_state=42)
```

BoW 결과

6000×5000 특징 행렬: 행(리뷰), 열(특징벡터)

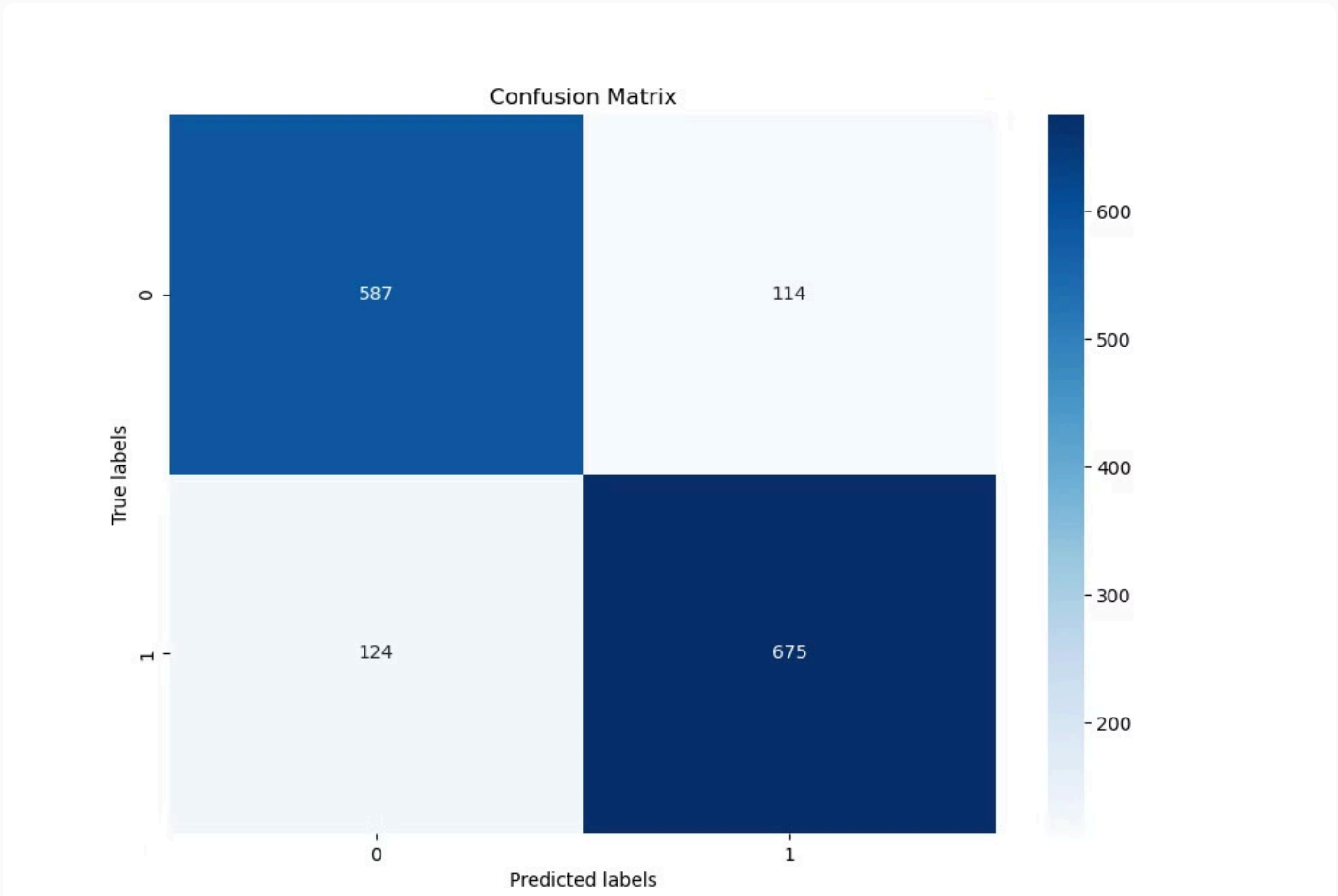
	0	1	2	3	4	5	6	7	8
54	0	0	0	0	0	0	0	1	0
55	0	0	0	0	0	0	0	0	0
56	0	0	0	0	0	0	0	0	0
57	0	0	0	0	0	0	0	0	0
58	0	0	0	0	0	0	0	0	0
59	0	0	0	0	0	0	0	0	0
60	0	0	0	0	0	0	0	0	0
61	0	0	0	0	0	0	1	0	0
62	0	0	0	0	0	0	0	0	0
63	0	0	0	0	0	0	0	0	0
64	0	0	0	0	0	0	0	0	0
65	0	0	0	0	0	0	0	0	0
66	0	0	0	0	0	1	0	0	0
67	0	0	0	0	0	0	0	0	0

로지스틱 회기 - 모델 구축

```
learning_rate=0.01
learning_epochs=20
batch_size = 4

model = models.Sequential([
    layers.Input(shape=(x_train.shape[1],)),
    layers.Dense(1, activation='sigmoid')
])
optimizer = SGD(learning_rate=learning_rate)
model.compile(optimizer=optimizer, loss='binary_crossentropy', metrics=['accuracy'])
history = model.fit(x_train, y_train, epochs=learning_epochs,
                    batch_size=batch_size, validation_data=(x_test, y_test))

model.save("C:\\Users\\ajm\\Desktop\\MyDir\\logistic_review.keras")
model = load_model("C:\\Users\\ajm\\Desktop\\MyDir\\logistic_review.keras")
```

로지스틱 회기 - 혼동행렬 코드

```
predictions = model.predict(x_test)
y_pred = [1 if pred > 0.5 else 0 for pred in predictions]
y_pred = np.array(y_pred)
cm = confusion_matrix(y_test, y_pred)
plt.figure(figsize=(10, 7))
sns.heatmap(cm, annot=True, fmt="d", cmap="Blues")
plt.xlabel('Predicted labels')
plt.ylabel('True labels')
plt.title('Confusion Matrix')
```



분류기 성능 평가

ROC 곡선

sklearn의 roc_curve 함수를 사용하여 위양성률과 진양성률을 계산.

시각화

Matplotlib을 사용하여 ROC 곡선을 그려 기준선과 비교.

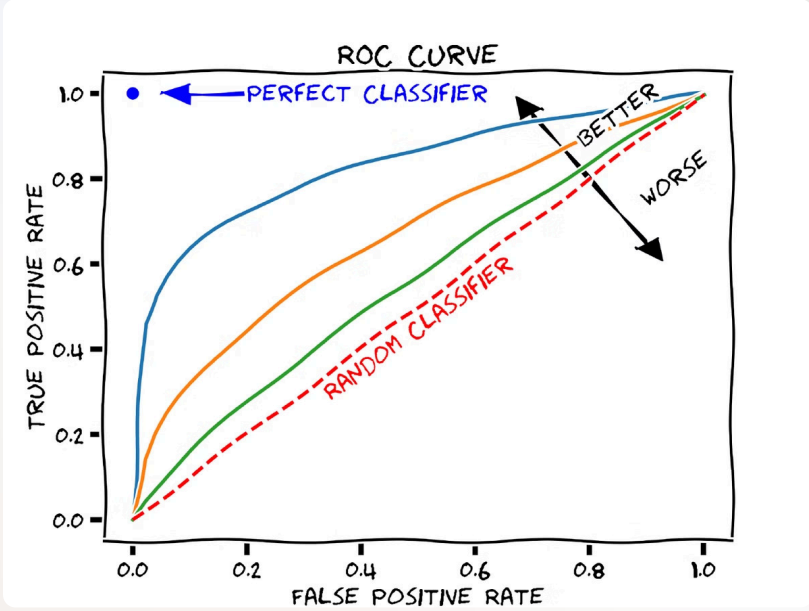
1

2

3

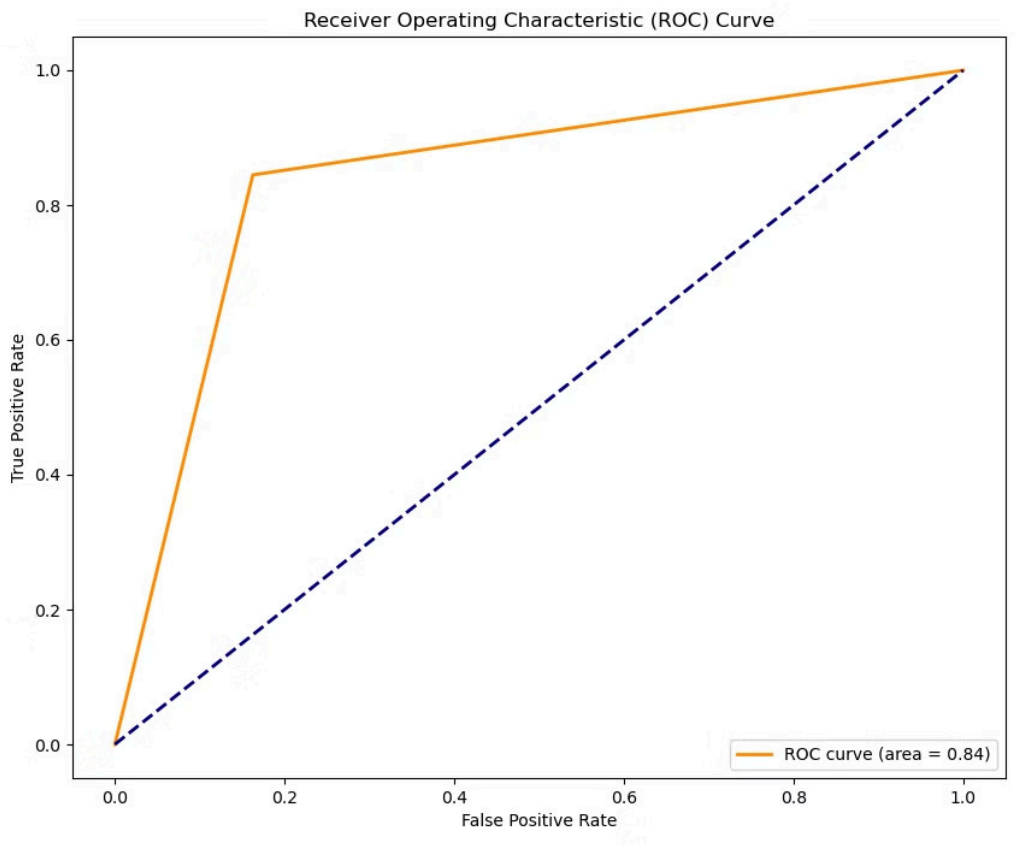
AUC

곡선 아래 면적(AUC)을 계산하여 전반적인 분류기 성능을 측정.



로지스틱 회기 - ROC코드 결과

```
predictions = model.predict(x_test)
y_pfpr, tpr, thresholds = roc_curve(y_test, y_pred)
roc_auc = auc(fpr, tpr)
plt.figure(figsize=(10, 8))
plt.plot(fpr, tpr, color='darkorange', lw=2, label=f'ROC curve (area = {roc_auc:.2f})')
plt.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--')
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Receiver Operating Characteristic (ROC) Curve')
plt.legend(loc="lower right")
plt.show()
```



소프트맥스 회귀 분류기

입력 데이터 준비

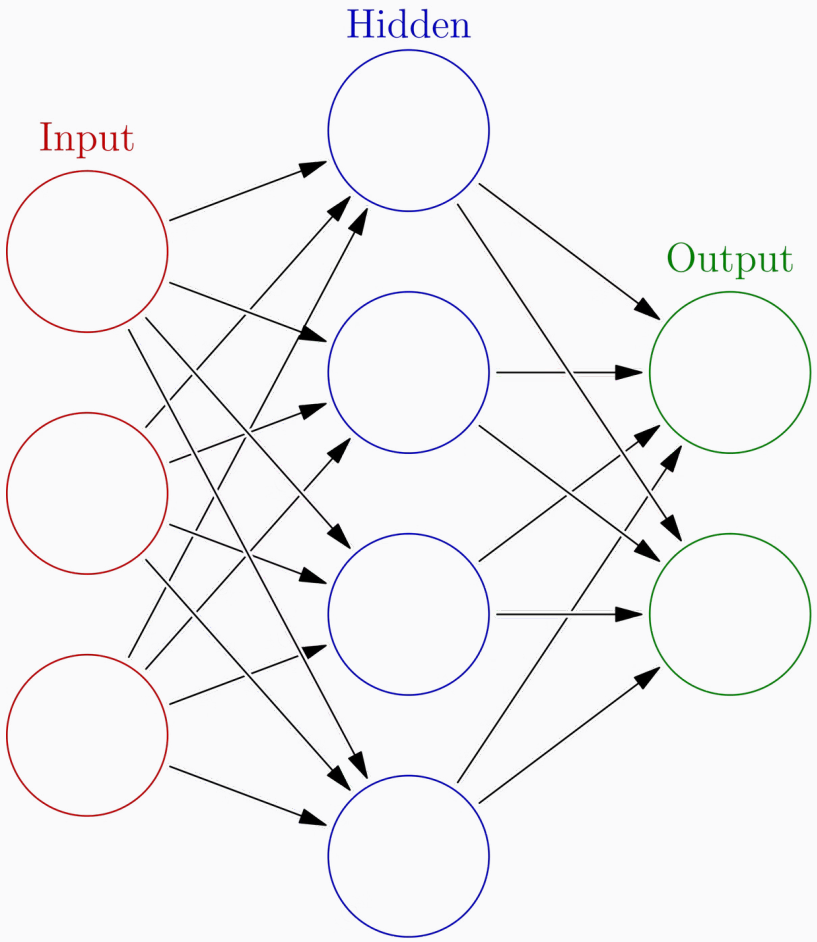
원-핫 인코딩된 레이블을 생성.

모델 구성

활성함수 - 소프트맥스 , 비용함수-categorical_crossentropy

훈련

경사 하강 최적화를 사용한 배치 훈련 수행.



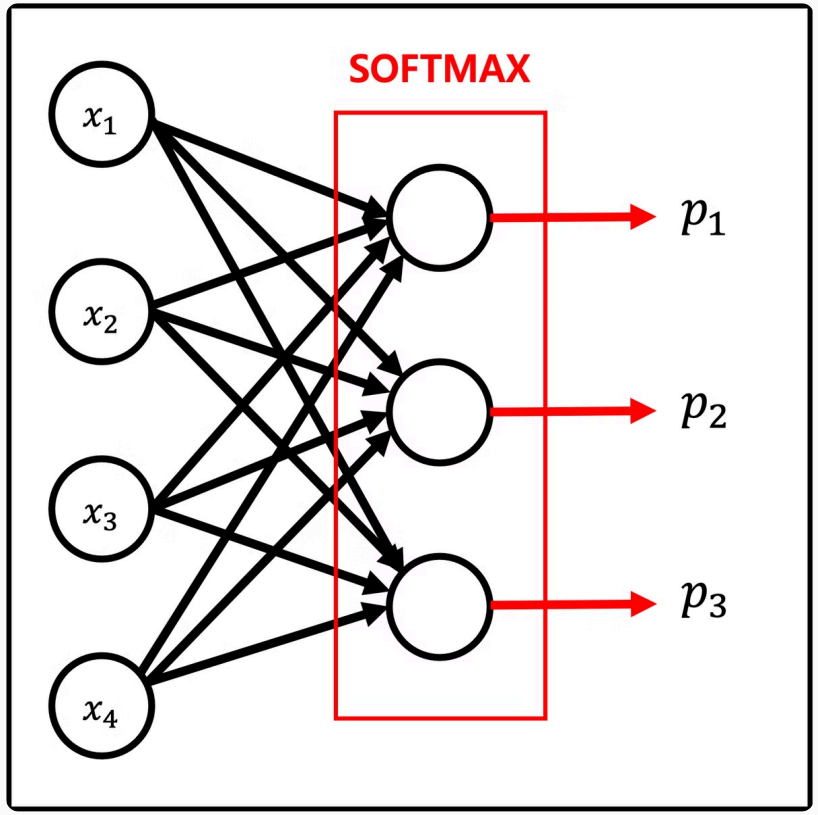
소프트맥스 회귀 모델 구조

입력 레이어는 5,000차원 Bag of Words 벡터를 가진 25,000개의 리뷰로 구성.

가중치 행렬 W는 긍정 및 부정 클래스에 대해 (6000, 2) 크기.

편향 벡터 b는 긍정 및 부정 클래스에 대해 (2) 크기.

출력은 긍정 및 부정 감성에 대한 소프트맥스 확률.



소프트맥스 회기-입력데이터 One-Hot Encoding 코드

```
clean_train_reviews = []
for i in range(0, x_train_short["review"].size):
    clean_train_reviews.append(review_to_words(x_train_short["review"][i]))

vectorizer = CountVectorizer(analyzer='word',max_features = 5000)
x_train = vectorizer.fit_transform(clean_train_reviews).toarray()
x_train, x_test, y_train, y_test = train_test_split(x_train, y_train, test_size=0.3, random_state=42)

encoder = OneHotEncoder(sparse_output=False)
y_train = encoder.fit_transform(y_train.reshape(-1,1))
y_test = encoder.fit_transform(y_test.reshape(-1,1))
```

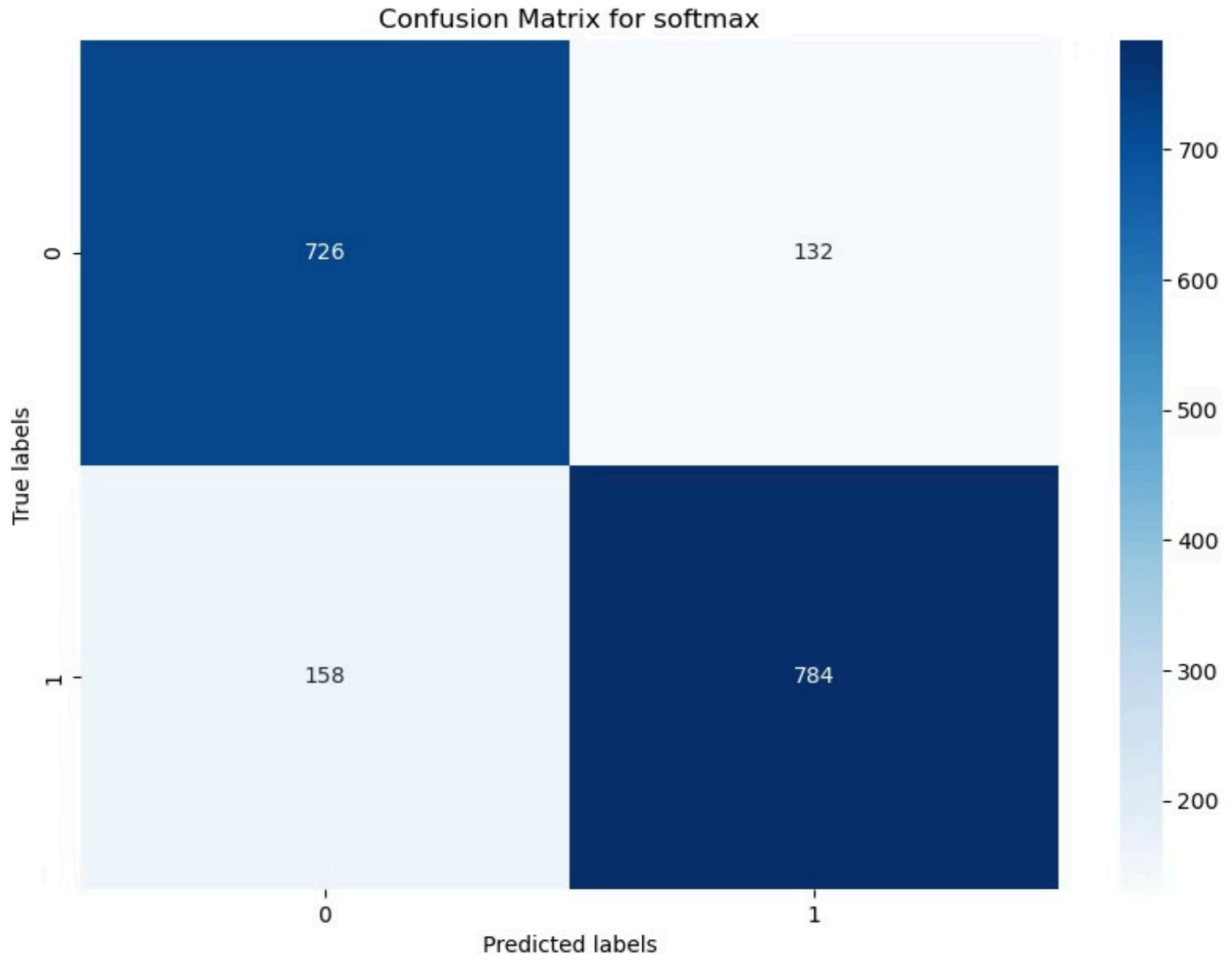
소프트맥스 회기 - 모델 코드

```
x_train = tf.convert_to_tensor(x_train,dtype=tf.float32)
y_train = tf.convert_to_tensor(y_train,dtype=tf.float32)
x_test = tf.convert_to_tensor(x_test,dtype=tf.float32)
y_test = tf.convert_to_tensor(y_test,dtype=tf.float32)

learning_rate=0.01
learning_epochs=20
batch_size = 4

model = models.Sequential([
    layers.Input(shape=(5000,)),
    layers.Dense(2, activation='softmax')
])
optimizer = SGD(learning_rate=learning_rate)
model.compile(optimizer=optimizer, loss='categorical_crossentropy', metrics=['accuracy'])
history = model.fit(x_train, y_train, epochs=learning_epochs,
                    batch_size=batch_size, validation_data=(x_test, y_test))
```

소프트맥스 회기 - 혼동 행렬 코드

```
predictions = model.predict(x_test)
y_pred = np.argmax(predictions,axis=1)
y_pred = np.array(y_pred)
y_test_n = np.argmax(y_test,axis=1)
cm = confusion_matrix(y_test_n, y_pred)
plt.figure(figsize=(10, 7))
sns.heatmap(cm, annot=True, fmt="d", cmap="Blues")
plt.xlabel('Predicted labels')
plt.ylabel('True labels')
plt.title('Confusion Matrix for softmax')
```

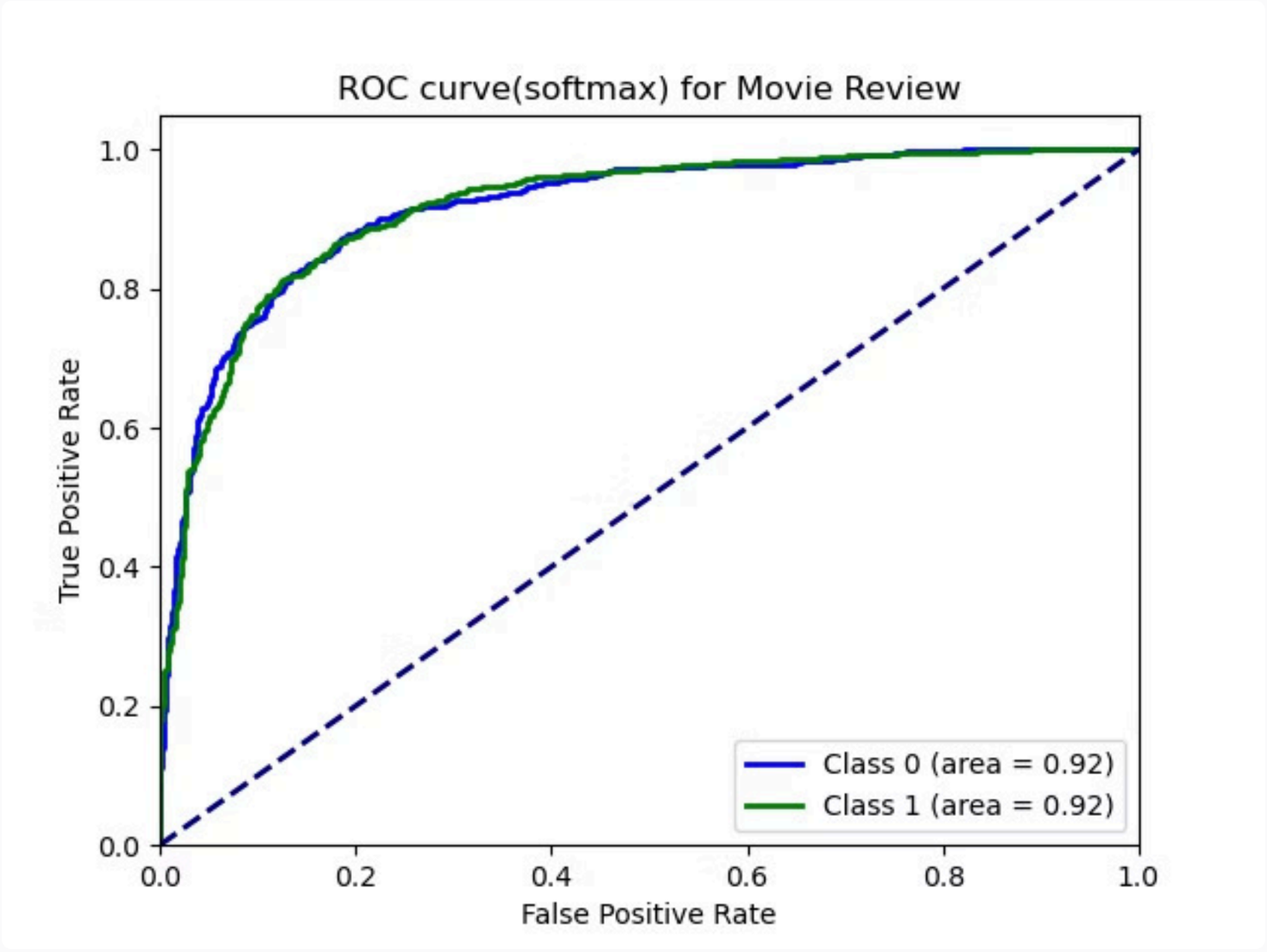
소프트맥스 회기 - 혼동 행렬 결과



소프트맥스 회기-ROC코드

```
y_pred_prob = model.predict(x_test)
fpr = {}
tpr = {}
roc_auc = {}
for i in range(2):  # 클래스가 2개이므로 0, 1에 대해 계산
    fpr[i], tpr[i], _ = roc_curve(y_test[:, i], y_pred_prob[:, i])
    roc_auc[i] = auc(fpr[i], tpr[i])

# ROC 곡선 그리기
plt.figure()
colors = ['blue', 'green']
for i in range(2):
    plt.plot(fpr[i], tpr[i], color=colors[i], lw=2, label=f'Class {i} (area = {roc_auc[i]:.2f})')
plt.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--')
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('ROC curve(softmax) for Movie Review')
plt.legend(loc="lower right")
plt.show()
```

소프트맥스 회기-ROC그래프



결론-분류기 성능 비교

분류기	정확도	AUC
로지스틱 회귀	83.9%	0.84
소프트맥스 회귀	83.5%	0.92

AUC는 모델이 양성 클래스와 음성 클래스를 얼마나 잘 구분하는지 평가.

AUC는 **불균형 데이터**에서 유용, 즉 "1"인 라벨과 "0"인 라벨이 골고루 분포되어 있지 않을 경우, 모델이 클래스 간의 구분을 얼마나 잘하는지 평가.

정확도는 클래스가 불균형일 때 잘못된 성능 평가를 제공함. 예를 들어, 전체의 90%가 음성 클래스라면, 모든 샘플을 음성으로 예측해도 정확도가 90%임.

소프트맥스 회귀가 더 높은 AUC를 보여주며, 감성 분류 작업에서 더 효과적임.

정확도 및 AUC

$$\text{정확도} = \frac{TP + TN}{TP + TN + FP + FN}$$

$$\text{정밀도} = \frac{TP}{TP + FP}$$

$$\text{재현율} = \frac{TP}{TP + FN}$$