📚 Day 3 함수 정의서: NSMC 데이터 분석 함수들

📋 개요

본 문서는 `data_overview.ipynb` 노트북에서 사용된 모든 함수들의 정의, 매개변수, 반환값, 사용 목적을 체계적으로 정리한 참조 문서입니다.

🔧 데이터 로딩 및 기본 분석 함수들

1. `load_nsmc_data()`

목적: NSMC 데이터를 안전하게 로딩하는 함수

```
python

def load_nsmc_data():
    """NSMC 데이터를 안전하게 로딩하는 함수 (경로 자동 탐지)"""
```

매개변수: 없음

반환값:

- `train_df` (pandas.DataFrame): 훈련 데이터 (150,000개 행)
- `test_df` (pandas.DataFrame): 테스트 데이터 (50,000개 행)

주요 기능:

- 다양한 경로에서 NSMC 데이터 파일 자동 탐지
- 여러 인코딩(utf-8, cp949, euc-kr) 시도하여 안정적 로딩
- 파일 크기 및 기본 정보 출력
- 오류 처리 및 예외 상황 대응

사용 예시:

```
python

train_df, test_df = load_nsmc_data()
```

핵심 로직:

1. 가능한 데이터 경로 리스트 순회
2. 파일 존재 여부 확인
3. 다양한 인코딩으로 로딩 시도

4. 성공시 기본 정보 출력 및 반환

---

2. `explore_data_structure(df, dataset_name)`

목적: 데이터 구조를 체계적으로 탐색하는 함수

```python
def explore_data_structure(df, dataset_name):
    """데이터 구조를 체계적으로 탐색하는 함수"""
```

매개변수:

- `df` (pandas.DataFrame): 분석할 데이터프레임
- `dataset_name` (str): 데이터셋 이름 (출력용)

반환값:

- `df.info()` 결과 (데이터프레임 정보)

주요 기능:

- 기본 정보 출력 (행/열 개수, 메모리 사용량)
- 컬럼별 데이터 타입 및 결측값 확인
- 고유값 개수 계산
- 첫 5개 행 미리보기
- 기본 통계 정보 출력

사용 예시:

```python
explore_data_structure(train_df, "훈련")
```

---

3. `check_data_quality(df, dataset_name)`

목적: 데이터 품질을 종합적으로 확인하는 함수

```python
def check_data_quality(df, dataset_name):
    """데이터 품질을 확인하는 함수"""
```

매개변수:

- `df` (pandas.DataFrame): 검사할 데이터프레임
- `dataset_name` (str): 데이터셋 이름

반환값: 없음 (출력만 수행)

주요 기능:

- 결측값 현황 분석
- 중복 행 및 중복 ID 확인
- 빈 텍스트 검사
- 라벨 값의 유효성 확인
- 감정 라벨 분포 계산

핵심 검사 항목:

- 데이터 완정성 (결측값, 중복값)
- 텍스트 품질 (빈 문자열)
- 라벨 일관성 (0, 1 값 확인)

📏 텍스트 길이 분석 함수들

4. `analyze_text_length(train_df, test_df)`

목적: 리뷰 텍스트 길이를 종합적으로 분석하는 함수

```python
def analyze_text_length(train_df, test_df):
    """리뷰 텍스트 길이를 분석하는 함수"""
```

매개변수:

- `train_df` (pandas.DataFrame): 훈련 데이터
- `test_df` (pandas.DataFrame): 테스트 데이터

반환값:

- `train_df` (pandas.DataFrame): text_length 컬럼이 추가된 훈련 데이터
- `test_df` (pandas.DataFrame): text_length 컬럼이 추가된 테스트 데이터

주요 기능:

- 텍스트 길이 계산 (.str.len() 사용)
- 기본 통계량 출력 (평균, 중앙값, 표준편차 등)
- 감정별 길이 비교 분석
- 특별한 길이 정보 (최대/최소) 출력

생성되는 통계:

- 평균, 중앙값, 표준편차
- 사분위수 (25%, 50%, 75%)
- 감정별 길이 분포 차이

---

5. `analyze_length_distribution(train_df)`

목적: 텍스트 길이를 구간별로 분류하고 분포를 분석하는 함수

```python
def analyze_length_distribution(train_df):
    """텍스트 길이 구간별 분포를 분석하는 함수"""
```

매개변수:

- `train_df` (pandas.DataFrame): 분석할 훈련 데이터

반환값:

- `train_df` (pandas.DataFrame): length_category 컬럼이 추가된 데이터

주요 기능:

- 길이 구간 정의 (1-10자, 11-20자, 21-50자 등)
- 구간별 리뷰 개수 및 비율 계산
- 누적 분포 계산 (특정 길이 이하 비율)

구간 정의:

- 매우짧음 (1-10자)
- 짧음 (11-20자)
- 보통 (21-50자)
- 긴편 (51-100자)
- 매우김 (101-200자)
- 극도로김 (200자+)

---

6. `visualize_text_length(train_df, test_df)`

목적: 텍스트 길이를 다양한 방식으로 시각화하는 함수

```
python

def visualize_text_length(train_df, test_df):
    """텍스트 길이를 다양한 방식으로 시각화하는 함수"""
```

매개변수:

- `train_df` (pandas.DataFrame): 훈련 데이터
- `test_df` (pandas.DataFrame): 테스트 데이터

반환값: 없음 (시각화만 수행)

생성되는 그래프:

1. **전체 길이 분포 히스토그램**: 훈련/테스트 데이터 비교
2. **감정별 길이 분포**: 긍정/부정 리뷰 길이 비교
3. **박스플롯**: 감정별 길이 분포의 사분위수 표시
4. **원형 차트**: 길이 구간별 비율 시각화

시각화 특징:

- 한글 폰트 설정으로 텍스트 정상 표시
- 평균선, 중앙값선 표시
- 색상 구분으로 직관적 이해 도움

---

7. `show_extreme_length_reviews(train_df)`

목적: 극단적인 길이의 리뷰 실제 내용을 확인하는 함수

```
python

def show_extreme_length_reviews(train_df):
    """극단적인 길이의 리뷰들을 확인하는 함수"""
```

매개변수:

- `train_df` (pandas.DataFrame): 분석할 데이터

반환값: 없음 (출력만 수행)

주요 기능:

- 가장 짧은 리뷰 5개 출력
- 가장 긴 리뷰 3개 출력 (100자까지만 표시)
- 평균 길이 근처 리뷰 3개 출력

출력 정보:

- 리뷰 길이 (문자 수)
- 감정 라벨 (긍정/부정)
- 실제 리뷰 내용

🔤 한국어 텍스트 특성 분석 함수들

8. `analyze_text_characteristics(df, dataset_name)`

목적: 텍스트의 문자 유형별 특성을 분석하는 함수

```python
def analyze_text_characteristics(df, dataset_name):
    """텍스트의 문자 유형별 특성을 분석하는 함수"""
```

매개변수:

- `df` (pandas.DataFrame): 분석할 데이터
- `dataset_name` (str): 데이터셋 이름

반환값:

- `stats` (dict): 각 문자 유형별 포함 비율 딕셔너리

분석하는 문자 유형:

- 한글: `[가-힣]` 패턴
- 영어: `[a-zA-Z]` 패턴
- 숫자: `[0-9]` 패턴
- 특수문자: `[^\w\s]` 패턴
- 이모티콘: `[ㅋㅎㅠㅜㅡㅗㅛㅕㅑㅏㅓㅣㅐㅔ]+` 패턴

주요 기능:

- 각 패턴별 포함 비율 계산
- 평균 단어 수 계산

- 한국어 텍스트 특성 종합 분석

---

9. `analyze_word_frequency(df, top_n=20)`

목적: 전체 텍스트에서 자주 사용되는 단어를 분석하는 함수

```python
def analyze_word_frequency(df, top_n=20):
    """전체 텍스트에서 자주 사용되는 단어를 분석하는 함수"""
```

매개변수:

- `df` (pandas.DataFrame): 분석할 데이터
- `top_n` (int): 출력할 상위 단어 개수 (기본값: 20)

반환값:

- `word_freq` (collections.Counter): 단어별 빈도수 Counter 객체

주요 기능:

- 모든 텍스트 합치기
- 한글 포함 단어만 필터링 (길이 2 이상)
- Counter를 사용한 빈도 계산
- 상위 N개 단어 출력

처리 과정:

1. 전체 텍스트 결합
2. 공백 기준 단어 분리
3. 한글 포함 && 길이 2 이상 필터링
4. 빈도 계산 및 정렬

---

10. `analyze_sentiment_words(df, top_n=10)`

목적: 감정별 특징적인 단어들을 분석하는 함수

```python
def analyze_sentiment_words(df, top_n=10):
    """감정별 특징적인 단어들을 분석하는 함수"""
```

매개변수:

- `df` (pandas.DataFrame): 분석할 데이터
- `top_n` (int): 출력할 상위 단어 개수 (기본값: 10)

반환값:

- `pos_freq` (collections.Counter): 긍정 리뷰 단어 빈도
- `neg_freq` (collections.Counter): 부정 리뷰 단어 빈도

주요 기능:

- 긍정/부정 리뷰 분리
- 각각의 단어 빈도 분석
- 공통 단어들의 감정별 사용 비율 비교
- 감정 구분력이 높은 단어 식별

비교 분석 단어: ['영화', '정말', '재미', '너무', '좋다', '별로', '최고', '최악', '진짜', '완전']

---

11. `analyze_special_patterns(df)`

목적: 이모티콘과 특수 표현 패턴을 분석하는 함수

```python
def analyze_special_patterns(df):
    """이모티콘과 특수 표현 패턴을 분석하는 함수"""
```

매개변수:

- `df` (pandas.DataFrame): 분석할 데이터

반환값: 없음 (출력만 수행)

분석하는 패턴들:

- ㅋㅋ (웃음): `ㅋ{2,}`
- ㅎㅎ (미소): `ㅎ{2,}`
- ㅠㅠ (슬픔): `ㅠ{2,}`
- ㅜㅜ (눈물): `ㅜ{2,}`
- !!! (강조): `!{2,}`
- ??? (의문): `\?{2,}`
- ... (여운): `\.{3,}`

주요 기능:

- 각 패턴의 전체 사용 빈도 계산
- 감정별 사용 비율 분석
- 이모티콘의 감정 예측력 평가

---

12. `show_special_text_examples(df)`

목적: 특별한 특성을 가진 텍스트 예시들을 보여주는 함수

```python
def show_special_text_examples(df):
    """특별한 특성을 가진 텍스트 예시들을 보여주는 함수"""
```

매개변수:

- `df` (pandas.DataFrame): 분석할 데이터

반환값: 없음 (출력만 수행)

보여주는 예시들:

- 이모티콘이 많은 리뷰 (3개 이상 연속)
- 영어가 포함된 리뷰
- 숫자가 포함된 리뷰
- 특수문자가 많은 리뷰 (5개 이상)

주요 기능:

- 특별한 패턴을 가진 리뷰 필터링
- 실제 사용 예시 출력
- 한국어 텍스트의 다양성 확인

📊 함수 활용 패턴

순차적 실행 구조

```python
```

```
# 1. 데이터 로딩
train_df, test_df = load_nsmc_data()

# 2. 기본 구조 분석
explore_data_structure(train_df, "훈련")
check_data_quality(train_df, "훈련")

# 3. 텍스트 길이 분석
train_df, test_df = analyze_text_length(train_df, test_df)
train_df = analyze_length_distribution(train_df)
visualize_text_length(train_df, test_df)

# 4. 한국어 특성 분석
analyze_text_characteristics(train_df, "훈련")
analyze_word_frequency(train_df)
analyze_sentiment_words(train_df)
analyze_special_patterns(train_df)
```

함수 설계 원칙

1. 단일 책임 원칙

- 각 함수는 하나의 명확한 분석 목적
- 기능별 분리로 재사용성 향상

2. 일관된 인터페이스

- 매개변수 명명 규칙 통일
- 출력 형식 표준화

3. 에러 처리

- 예외 상황 대응
- 안전한 데이터 처리

4. 문서화

- 독스트링으로 함수 목적 명시
- 매개변수와 반환값 설명

🎯 함수 활용 시 주의사항

데이터 의존성

- `text_length` 컬럼이 필요한 함수들은 `analyze_text_length()` 실행 후 사용

- `length_category` 컬럼이 필요한 경우 `analyze_length_distribution()` 선행 실행

메모리 관리

- 대용량 텍스트 처리 시 메모리 사용량 고려
- 필요시 샘플링으로 메모리 절약

한글 폰트 설정

- 시각화 함수 실행 전 한글 폰트 설정 필수
- `plt.rcParams['font.family'] = 'Malgun Gothic'`

정규표현식 성능

- 복잡한 패턴 매칭 시 처리 시간 고려
- 필요시 패턴 컴파일로 성능 최적화

이 함수 정의서는 코드 재사용, 유지보수, 그리고 추후 확장 개발 시 참조 문서로 활용할 수 있습니다.