🎬 NSMC 기반 한국어 영화 리뷰 감정분석 프로젝트 계획서

📋 프로젝트 개요

프로젝트명

NSMC 기반 한국어 영화 리뷰 감정분석 시스템 개발

개발 방법론 적용

- **프로젝트 구조**: 체계적인 디렉토리 구성 및 명명 규칙
- **코딩 품질**: Python 베스트 프랙티스 적용
- **실험 관리**: 모델 성능 추적 및 비교
- **문서화**: 재사용 가능한 코드 및 설명서 작성

데이터 정보

- **사용 데이터**: NSMC (Naver Sentiment Movie Corpus)
- **데이터 출처**: https://github.com/e9t/nsmc
- **데이터 형태**: TSV 형태의 텍스트 파일 (.txt 확장자)
- **데이터 규모**: 총 200,000개 리뷰 (훈련 150k + 테스트 50k)
- **라벨 구성**: 긍정(1)/부정(0) 완벽한 50:50 균형

프로젝트 목적

- **학습 목표**:
  - 한국어 자연어처리(NLP) 실무 경험 및 포트폴리오 구축
  - 체계적인 ML 프로젝트 관리 및 개발 프로세스 학습
  - 실험 추적 및 결과 분석 능력 향상
- **기술 목표**:
  - NSMC 테스트 데이터 기준 90% 이상 정확도 달성
  - 재현 가능한 ML 파이프라인 구축
- **포트폴리오 목표**:
  - 실무 수준의 한국어 NLP 프로젝트 완성
  - 기업에서 인정받을 수 있는 코드 품질 달성

프로젝트 범위

- **핵심 작업**:
  - NSMC 데이터 분석 및 전처리
  - 한국어 특화 NLP 파이프라인 구축

- 다양한 모델 구현 및 성능 비교 (기본 ML → 딥러닝 → Transformer)
- 성능 평가 및 결과 해석
- 웹 데모 및 포트폴리오 완성

- **제외 사항**:
  - 대용량 실시간 처리
  - 상용 서비스 운영
  - 다국어 지원 (한국어 집중)
  - 추가 데이터 수집

🎯 프로젝트 목표 및 성공 지표

주요 목표 (SMART 기준)

1. 기술적 목표

- **목표**: NSMC 테스트 데이터셋 기준 90% 이상 정확도 달성
- **측정 기준**: Test Set Accuracy, F1-Score, Precision, Recall
- **달성 기한**: 8주차 말까지
- **실현 가능성**: 기존 NSMC 연구 결과 85-95% 수준으로 달성 가능
- **세부 목표**:
  - 기본 ML 모델: 80% 이상
  - 딥러닝 모델: 85% 이상
  - Transformer 모델: 90% 이상

2. 학습 목표

- **목표**: NLP 전 과정 + 체계적 개발 프로세스 경험
- **측정 기준**: 각 단계별 산출물 완성도
- **달성 기한**: 10주차 말까지
- **실현 가능성**: 단계별 점진적 학습으로 달성 가능

3. 포트폴리오 목표

- **목표**: GitHub 스타 10개 이상, 기업급 품질 인정
- **측정 기준**: GitHub 활동 지표, 코드 품질 점수
- **달성 기한**: 프로젝트 완료 후 2주 내
- **실현 가능성**: 체계적 개발 프로세스로 고품질 확보

핵심 성과 지표 (KPI)

구분	지표	목표값	측정 방법
기술	모델 정확도	90%+	NSMC 테스트셋 평가
기술	F1-Score	0.90+	분류 성능 측정
효율성	추론 속도	1초/1000건	처리 시간 측정
코드품질	가독성 점수	우수	코드 리뷰 기준
완성도	문서화 완성도	100%	체크리스트 달성
포트폴리오	GitHub 관심도	10+ 스타	프로젝트 인지도

📊 이해관계자 분석

주요 이해관계자

1. **프로젝트 수행자 (본인)**
   - 역할: 전체 프로젝트 기획, 개발, 실행
   - 기대사항: 포트폴리오 완성, 기술 역량 향상, 체계적 개발 프로세스 체득
   - 영향도: 최고
2. **잠재적 채용 담당자**
   - 역할: 포트폴리오 평가자
   - 기대사항: 실무 적용 가능한 기술 수준 + 체계적 개발 프로세스 이해
   - 영향도: 높음
3. **NLP 커뮤니티**
   - 역할: 피드백 제공자, 지식 공유 파트너
   - 기대사항: 재사용 가능한 코드, 명확한 문서화
   - 영향도: 중간

의사소통 계획

- **진행 상황 공유**: 주간 GitHub 커밋, 블로그 포스팅
- **피드백 수집**: NLP 관련 온라인 커뮤니티 활용
- **멘토링**: 필요시 전문가 멘토링 요청

⏰ 프로젝트 일정 및 마일스톤

전체 일정

- **시작일**: 2025년 X월 X일

- **종료일**: 2025년 X월 X일 (총 10주)
- **주간 투입 시간**: 10-15시간

단계별 마일스톤

Phase 1: 기초 설정 및 데이터 이해 (1-2주)

목표: NSMC 데이터 파악 및 프로젝트 환경 구축

1주차 - 환경 설정

- [ ] Python 환경 및 필수 라이브러리 설치
- [ ] 깔끔한 프로젝트 폴더 구조 만들기
- [ ] NSMC 데이터 다운로드 및 첫 로딩 테스트
- [ ] GitHub 저장소 생성 및 첫 커밋

2주차 - 데이터 탐색

- [ ] 200K 리뷰 데이터 기본 분석 (EDA)
- [ ] 긍정/부정 분포 확인 및 시각화
- [ ] 한국어 텍스트 특성 파악 (길이, 특수문자 등)
- [ ] KoNLPy 도구 설치 및 토큰화 테스트

주요 결과물:

- 체계적인 프로젝트 구조
- 데이터 로딩 및 기본 분석 스크립트
- EDA 결과 노트북
- 한국어 텍스트 특성 분석 보고서

Phase 2: 기본 모델 개발 (3-5주)

목표: 전통적인 ML 기법으로 첫 번째 성능 기준 설정

3주차 - 데이터 전처리

- [ ] 한국어 텍스트 정제 및 토큰화 파이프라인 구축
- [ ] Bag of Words, TF-IDF 특성 추출
- [ ] 첫 번째 모델 (로지스틱 회귀, SVM) 구현

4주차 - 모델 최적화

- [ ] 하이퍼파라미터 튜닝 및 교차 검증
- [ ] 성능 평가 및 결과 분석
- [ ] 첫 번째 벤치마크 설정 (목표: 80%+ 정확도)

5주차 - 성능 개선

- [ ] 앙상블 모델 시도 (Voting, Bagging)
- [ ] 특성 엔지니어링 및 최적화
- [ ] 오류 분석 및 개선점 도출

주요 결과물:

- 한국어 전처리 모듈
- 기본 ML 모델들 (80%+ 정확도)
- 성능 비교 및 분석 보고서

Phase 3: 딥러닝 모델 개발 (6-7주)

목표: 딥러닝으로 성능 향상 및 85%+ 정확도 달성

6주차 - 딥러닝 기초

- [ ] Word2Vec, FastText 임베딩 구현
- [ ] LSTM, CNN 텍스트 분류 모델 개발
- [ ] GPU 활용 및 학습 최적화

7주차 - 성능 최적화

- [ ] 모델 앙상블 및 하이퍼파라미터 튜닝
- [ ] 조기 종료 및 모델 체크포인트 적용
- [ ] 성능 모니터링 및 결과 분석

주요 결과물:

- 딥러닝 모델 파이프라인
- 임베딩 벡터 분석 결과
- 성능 개선 분석 보고서

Phase 4: Transformer 모델 (8주)

목표: 최신 기법으로 90%+ 정확도 달성

8주차 - 최고 성능 모델

- [ ] KoBERT, KoELECTRA 파인튜닝
- [ ] 모델 해석 및 어텐션 시각화
- [ ] 최종 모델 선택 및 성능 검증

주요 결과물:

- 최고 성능 Transformer 모델
- 모델 해석 및 분석 결과

Phase 5: 웹 데모 및 마무리 (9-10주)

목표: 포트폴리오 완성 및 프로젝트 정리

9주차 - 웹 데모 개발

- [ ] Streamlit 기반 감정분석 웹 앱 개발
- [ ] 모델 서빙 및 실시간 예측 구현
- [ ] 사용자 인터페이스 완성

10주차 - 프로젝트 완성

- [ ] GitHub README 및 문서 정리
- [ ] 기술 블로그 포스팅 작성
- [ ] 프로젝트 발표 자료 준비

주요 결과물:

- 완성된 웹 데모
- 포트폴리오급 문서
- 기술 블로그 포스팅

💰 리소스 요구사항

인적 리소스

- **주 수행자**: 1명 (본인)
- **필요 역량**: Python 기초, 머신러닝 기본 지식
- **추가 학습 필요**: 한국어 NLP, 딥러닝, 웹 개발 기초

기술적 리소스

하드웨어

- **CPU**: Intel i5 이상 (또는 동급)
- **RAM**: 16GB 이상 (딥러닝 모델 학습 고려)
- **Storage**: 50GB 이상 여유 공간 (모델 실험 및 버전 관리 고려)
- **GPU**: 선택사항 (클라우드 대안 가능, Transformer 모델 학습 시 필요)

소프트웨어 (추천 도구)

- **개발 환경**: Python 3.8+, Jupyter Notebook, VS Code
- **필수 라이브러리**:
  - 기본: pandas, numpy, matplotlib, seaborn

    - NLP: konlpy, transformers, gensim, nltk
    - ML/DL: scikit-learn, tensorflow, torch
- **선택 도구**:
    - 실험 관리: mlflow (성능 추적용)
    - 코드 품질: black (코드 포맷팅)
    - 버전 관리: Git, GitHub
- **클라우드**: Google Colab (무료 GPU), Kaggle Kernels

데이터

주 데이터: NSMC (Naver Sentiment Movie Corpus)

- **파일 구성**:
    - `ratings_train.txt`: 150,000개 훈련 리뷰
    - `ratings_test.txt`: 50,000개 테스트 리뷰
    - `ratings.txt`: 전체 200,000개 리뷰
- **파일 형태**: TSV (탭 구분) 텍스트 파일
- **컬럼 구성**: id, document, label
- **라벨**: 0(부정), 1(긍정) - 완벽한 50:50 균형
- **특성**: 모든 리뷰 140자 이하, 중립 리뷰(5-8점) 제외
- **다운로드**: GitHub (https://github.com/e9t/nsmc) - 무료
- **용량**: 약 25MB (압축 해제 후)

보조 데이터

- **한국어 사전 훈련 모델**: KoBERT, KoELECTRA (Hugging Face)
- **한국어 임베딩**: FastText Korean (무료)

예산

- **총 예산**: 0-30달러 (월별, 선택사항)
- **필수 비용**: 0원 (무료 도구만으로 완성 가능)
- **선택적 투자**:
    - Google Colab Pro: 월 10달러 (GPU 가속)
    - 도메인 + 호스팅: 월 5-10달러 (웹 데모용)
    - 유료 멘토링: 회당 5-10만원 (선택사항)

⚠ 리스크 관리

주요 리스크 및 대응 방안

1. 기술적 리스크

리스크: 한국어 NLP 복잡성 및 NSMC 텍스트 데이터 특성으로 인한 진행 지연

- **확률**: 중간 (40%)
- **영향도**: 높음
- **구체적 이슈**:
  - TSV 파일 인코딩 문제
  - 한국어 띄어쓰기, 맞춤법 오류
  - 구어체, 슬랭, 줄임말 처리 어려움
- **대응 방안**:
  - 단계별 점진적 접근 (간단한 전처리부터)
  - KoNLPy 커뮤니티 및 NSMC 사용 사례 적극 활용
  - 필요시 영어 IMDb 데이터로 대안 실습 후 한국어 적용

리스크: NSMC 기준 목표 성능(90%) 미달

- **확률**: 낮음 (20%)
- **영향도**: 중간
- **구체적 이슈**:
  - 150K 훈련 데이터로 성능 한계
  - 50K 테스트 데이터에서 과적합
- **대응 방안**:
  - 현실적 목표 재설정 (85% → 80%)
  - 앙상블 기법 및 데이터 증강 적극 활용
  - KoBERT 등 사전 훈련 모델 활용

2. 일정 리스크

리스크: 학습 곡선으로 인한 일정 지연

- **확률**: 높음 (60%)
- **영향도**: 중간
- **구체적 이슈**:
  - 새로운 도구 학습 시간
  - 문서화 작업 시간
  - 디버깅 및 최적화 시간

- **대응 방안**:
  - 버퍼 시간 확보 (10주 → 12주)
  - 핵심 기능 우선 개발 (MVP 접근)
  - 주간 진도 체크 및 조정

3. 리소스 리스크

리스크: NSMC 200K 데이터 처리를 위한 컴퓨팅 자원 부족

- **확률**: 낮음 (30%)
- **영향도**: 중간
- **구체적 이슈**:
  - 150K 훈련 데이터 메모리 부족
  - 딥러닝 모델 학습 시간 과다
  - 대용량 임베딩 벡터 처리 한계
- **대응 방안**:
  - Google Colab Pro 활용 (GPU 메모리 확보)
  - 배치 처리 및 모델 크기 조정
  - 클라우드 플랫폼 무료 크레딧 적극 활용
  - 필요시 NSMC 부분 샘플링 (50K → 20K)

품질 보증 계획

- **코드 리뷰**: 주요 모듈별 자체 리뷰
- **성능 모니터링**: 단계별 성능 지표 추적
- **문서화**: 실시간 문서 업데이트
- **버전 관리**: Git을 통한 체계적 버전 관리

📈 성공 요소 및 기대 효과

핵심 성공 요소

1. **체계적 접근**: PM 관점 + 개발 방법론 적용
2. **점진적 학습**: 기초부터 고급까지 단계적 발전
3. **실무 연관성**: 실제 비즈니스 + 체계적 개발 프로세스
4. **차별화**: 한국어 NLP + 전문적 개발 프로세스

기대 효과

단기 효과 (프로젝트 완료 시점)

- **기술 역량**: 한국어 NLP + 체계적 개발 프로세스 전문 지식 확보
- **포트폴리오**: 기업급 완성도의 프로젝트 사례
- **네트워킹**: NLP 커뮤니티 내 인지도 상승

장기 효과 (6개월 후)

- **취업 경쟁력**: 실무 즉시 투입 가능한 데이터 사이언티스트
- **프로젝트 확장**: 추천 시스템, 챗봇 등 연관 프로젝트 가능
- **리더십**: 팀 프로젝트에서 개발 프로세스 리드 가능

📋 체크리스트 및 평가 기준

프로젝트 완료 체크리스트

핵심 완료 사항

- [ ] NSMC 데이터 성공적 처리 및 분석
- [ ] 5개 이상 모델 구현 및 성능 비교
- [ ] 테스트 데이터 기준 90% 이상 정확도 달성
- [ ] 한국어 리뷰 오류 분석 및 개선 방안 도출
- [ ] 실시간 감정분석 웹 데모 구현

문서화 완료 사항

- [ ] 프로젝트 사용법이 담긴 README 작성
- [ ] 각 단계별 Jupyter Notebook 정리
- [ ] 기술 블로그 포스팅 완성
- [ ] 프로젝트 발표 자료 준비

코드 품질 확인 사항

- [ ] 읽기 쉬운 코드 작성 (주석, 함수명)
- [ ] 모듈화된 코드 구조
- [ ] Git 커밋 히스토리 정리
- [ ] 재현 가능한 실험 환경 구축

평가 기준 (간소화)

항목	가중치	평가 방법
기술적 성과	40%	모델 성능, 정확도 달성

항목	가중치	평가 방법
프로젝트 완성도	25%	웹 데모, 문서화 품질
코드 품질	20%	가독성, 구조화 수준
창의성	15%	독창적 접근, 인사이트 도출

🔄 프로젝트 이후 계획

즉시 후속 작업 (완료 후 1개월)

- **포트폴리오 홍보**: LinkedIn, GitHub, 기술 커뮤니티
- **피드백 수집**: 멘토, 동료 개발자들로부터 리뷰
- **성과 분석**: 프로젝트 성공/실패 요인 분석

확장 프로젝트 (완료 후 3개월)

- **NSMC 확장**: 다중 분류 (긍정/중립/부정) - 중립 데이터 추가 수집
- **실시간 처리**: 네이버 영화 리뷰 실시간 크롤링 및 감정 분석
- **타 플랫폼**: CGV, 메가박스, 왓챠 리뷰로 도메인 확장
- **성능 개선**: NSMC 기준 95%+ 정확도 도전

장기 발전 방향 (완료 후 6개월)

- **상용 서비스**: 영화 추천 시스템과 결합한 감정 기반 서비스
- **연구 활동**: NSMC 개선 방안 논문 작성 또는 새로운 한국어 감정분석 데이터셋 구축
- **교육 활동**: NSMC 활용 한국어 NLP 튜토리얼 제작 및 강의

📞 연락처 및 승인

프로젝트 매니저

- **이름**: [본인 이름]
- **이메일**: [이메일 주소]
- **GitHub**: [GitHub 주소]

계획서 승인

- **작성일**: 2025년 X월 X일
- **검토자**: 본인
- **승인일**: 2025년 X월 X일
- **다음 리뷰 예정일**: 2주 후

품질 확인 사항

- [ ] 프로젝트 구조가 체계적으로 구성되었는가?
- [ ] 파일 명명이 일관성 있게 관리되는가?
- [ ] 코드가 읽기 쉽고 이해하기 쉬운가?
- [ ] 실험 결과가 체계적으로 기록되는가?
- [ ] 문서화가 충분히 작성되었는가?

🔧 프로젝트 구조 및 실행 가이드

추천 프로젝트 구조

```
nsmc_sentiment_analysis/
├── data/                # 데이터 저장
│   ├── raw/             # 원본 NSMC 데이터
│   ├── processed/       # 전처리된 데이터
│   └── models/          # 저장된 모델들
├── notebooks/           # 분석 노트북
│   ├── 01_data_exploration.ipynb
│   ├── 02_preprocessing.ipynb
│   ├── 03_baseline_models.ipynb
│   ├── 04_deep_learning.ipynb
│   └── 05_final_model.ipynb
├── src/                 # 소스 코드
│   ├── data_loader.py   # 데이터 로딩
│   ├── preprocessor.py  # 전처리
│   ├── models.py        # 모델 구현
│   └── utils.py         # 유틸리티
├── app/                 # 웹 데모
│   └── streamlit_app.py
├── requirements.txt     # 패키지 목록
├── README.md            # 프로젝트 설명
└── .gitignore           # Git 제외 파일
```

파일 명명 규칙

파일 유형	명명 예시
노트북	01_data_exploration.ipynb
Python 파일	nsmc_preprocessor.py
모델 파일	kobert_final_model.pkl
설정 파일	config.yaml

간단한 Git 사용법

```bash
# 프로젝트 시작
git init
git add .
git commit -m "프로젝트 초기 설정"

# 주요 단계별 커밋
git commit -m "데이터 탐색 완료"
git commit -m "기본 모델 구현"
git commit -m "딥러닝 모델 추가"
git commit -m "웹 데모 완성"
```

🎯 1주차 시작 가이드

Day 1: 환경 설정

```bash
# 1. 가상환경 생성
conda create -n nsmc_project python=3.8
conda activate nsmc_project

# 2. 기본 라이브러리 설치
pip install pandas numpy matplotlib seaborn jupyter
pip install konlpy scikit-learn
```

Day 2: 프로젝트 설정

```bash
# 1. 폴더 구조 만들기
mkdir nsmc_sentiment_analysis && cd nsmc_sentiment_analysis
mkdir -p data/{raw,processed,models} notebooks src app

# 2. 기본 파일 생성
touch requirements.txt README.md .gitignore
```

Day 3: 데이터 다운로드

```bash
```

```
# NSMC 데이터 다운로드
cd data/raw/
wget https://raw.githubusercontent.com/e9t/nsmc/master/ratings_train.txt
wget https://raw.githubusercontent.com/e9t/nsmc/master/ratings_test.txt
cd ../../
```

Day 4-5: 첫 번째 분석

- 첫 번째 노트북 작성: `01_data_exploration.ipynb`
- 데이터 로딩 및 기본 정보 확인
- 간단한 시각화 생성

본 계획서는 초급자도 쉽게 따라할 수 있도록 핵심 내용 위주로 작성되었으며, 프로젝트 진행 상황에 따라 유연하게 조정할 수 있습니다.