도서관 데이터 공유 시스템 개선방안 제안

[redacted] 김현승

Table of contents

01

프로젝트 초기 진행상황, 피보팅

처음에 계획했던 프로젝트에 대해 설명하고,
프로젝트 주제를 변경하게 된 경위에 대해 설명합니다.

프로젝트 초기 목적

작은 도서관은 공립도서관의 접근성이 떨어지는 지역에서 지역 **주민들의 최소한의 정보 요구를 충족**시키기 위해 건립되었습니다. 하지만, 최근 들어 작은 도서관 운영에 어려움이 생겼습니다.

- ➔ 작은 도서관 대상 **예산 대폭 감축**
- ➔ 일부 지역 **작은 도서관 예산 미배정**

작은 도서관의 예산이 줄어들며, 작은 도서관이 이전과 같은 수준으로 서비스를 제공하기 위해선 **예산의 집행 과정을 효율적**으로 변경할 필요가 있었습니다. 이를 해결하기 위해 작은 도서관의 **도서 대출 패턴을 분석**하여 더 적은 돈으로 **효율적으로 수서전략**을 수립할 수 있도록 돕는 것이 당초 프로젝트의 목적이었습니다.

WBS

작업 항목 구분			계획기간		진척률	산출물	3월				4월				5월				6월			
1	2	3	시작날짜	종료날짜	달성률		W1	W2	W3	W4	W5	W6	W7	W8	W9	W10	W11	W12	W13	W14	W15	W16
프로젝트 명			2025. 3. 3	2025. 6. 16	27.7%		3/10	3/17	3/24	3/31	4/7	4/14	4/21	4/28	5/5	5/12	5/19	5/26	6/2	6/9	6/16	6/23
1 프로젝트관리			2025. 3. 3	2025. 6. 16	47.5%																	
	1.1 계획수립		2025. 3. 3	2025. 4. 17	95.0%																	
		1.1.1 일정계획 수립	2025. 3. 3	2025. 4. 17	90.0%	WBS																
		1.1.2 프로젝트 계획서 작성	2025. 4. 11	2025. 4. 17	100.0%	프로젝트 계획서																
	1.2 프로젝트 완료		2025. 6. 9	2025. 6. 16	0.0%																	
		1.2.1 최종발표	2025. 6. 9	2025. 6. 13	0.0%	발표자료																
		1.2.2 프로젝트 완료	2025. 6. 16	2025. 6. 16	0.0%																	
2 자료수집			2025. 3. 11	2025. 4. 28	63.3%																	
	2.1 데이터셋 탐색		2025. 3. 11	2025. 3. 17	100.0%																	
	2.2 데이터셋 로그정의서 확인		2025. 3. 11	2025. 3. 17	100.0%																	
	2.3 데이터셋 다운로드 및 병합		2025. 4. 1	2025. 4. 21	100.0%	.parquet 데이터셋																
	2.4 데이터셋 메모리 최적화		2025. 4. 1	2025. 4. 21	80.0%																	
	2.5 데이터 결측치, 이상치 처리		2025. 4. 1	2025. 4. 21	0.0%																	
	2.6 온라인 서정 인기도서 스크롤링		2025. 4. 22	2025. 4. 28	0.0%	.json 데이터셋																
3 데이터 분석			2025. 4. 22	2025. 5. 26	0.0%																	
	3.1 EDA		2025. 4. 22	2025. 4. 28	0.0%																	
		3.1.1 전반적 데이터 확인, 통계치 확인	2025. 4. 22	2025. 4. 28	0.0%																	
		3.1.2 컬럼간 Correlationship 확인	2025. 4. 22	2025. 4. 28	0.0%																	
		3.1.3 Correlationship 0.7 이상 컬럼 대상 T-test	2025. 4. 22	2025. 4. 28	0.0%	EDA 결과																
	3.2 ML		2025. 4. 22	2025. 5. 4	0.0%																	
		3.2.1 AutoGluon : 머신러닝 기법 결정	2025. 4. 22	2025. 4. 28	0.0%																	
		3.2.2 Sklearn : 머신러닝 환경 구축	2025. 4. 22	2025. 5. 4	0.0%																	
		3.2.3 GridSearch / Optuna : 모델 정밀도 고도화	2025. 4. 29	2025. 5. 4	0.0%																	
	3.3 ML 모델 평가 및 보완		2025. 5. 5	2025. 5. 25	0.0%																	
		3.3.1 ML 모델 평가 방법 결정 (F1-score / ROC curve)	2025. 5. 5	2025. 5. 11	0.0%																	
		3.3.2 ML 평가	2025. 5. 5	2025. 5. 25	0.0%																	
		3.3.3 ML 보완 : 데이터 전처리 고도화 / Optuna 반복시도	2025. 5. 5	2025. 5. 25	0.0%	ML 모델																
4 액션플랜 제시			2025. 5. 26	2025. 6. 16	0.0%																	
	4.4 EDA 결과를 통한 액션플랜 제시		2025. 5. 26	2025. 6. 2	0.0%	액션플랜																
	4.5 ML 모델 활용방안 제시		2025. 5. 26	2025. 6. 16	0.0%																	
		4.5.1 ML 모델 활용방안 구체화	2025. 5. 26	2025. 6. 2	0.0%																	
		4.5.2 ML 모델을 통한 직관적 액션플랜 제시	2025. 5. 26	2025. 6. 16	0.0%	액션플랜																
5 콘텐츠 구현			2025. 5. 26	2025. 6. 16	0.0%																	
	5.1 대시보드		2025. 5. 26	2025. 6. 16	0.0%																	
		5.1.1 핵심(우선순위)지표 선정	2025. 5. 26	2025. 6. 2	0.0%																	
		5.1.2 대시보드 레이아웃 결정	2025. 5. 26	2025. 6. 2	0.0%																	
		5.1.3 표본샘플 통한 레이아웃 제작	2025. 5. 26	2025. 6. 16	0.0%	대시보드 레이아웃																
		5.1.4 Tableau 대시보드 구현	2025. 5. 26	2025. 6. 16	0.0%	대시보드																
		5.1.5 대시보드 유지보수 가이드라인 제작	2025. 6. 3	2025. 6. 16	0.0%	대시보드 가이드라인																
	5.2 메일링 시스템		2025. 6. 3	2025. 6. 16	0.0%																	
		5.2.1 메일링 시스템 프레임워크 제작	2025. 6. 3	2025. 6. 9	0.0%																	
		5.2.2 라이프 데이터 용 핵심지표 선정	2025. 6. 3	2025. 6. 9	0.0%																	
		5.2.3 라이브 데이터 메일링 시스템 구현	2025. 6. 3	2025. 6. 16	0.0%	메일링 시스템																
		5.2.4 메일링 시스템 유지보수 가이드라인 제작	2025. 6. 3	2025. 6. 16	0.0%	메일링 시스템 가이드라인																

WBS 계획

데이터를 **수집**하고, 데이터 분석에 사용할 수 있도록 **전처리 과정**을 거친 다음, **EDA**, **머신러닝**을 통해 데이터 분석 과정을 거쳤습니다.
분석 결과를 바탕으로, **메일링 시스템**과 **실시간 반영 데이터 대시보드**까지 제작하는 것을 처음 프로젝트의 목표로 잡았습니다.

프로젝트 초기 진행상황 – 데이터 수집, 전처리

```python
# JSON 파일 디렉토리 설정
json_dir = "data/sml_lib_project"

# CSV 파일 내보낼 디렉토리 설정
output_files = {
    "file1": "data/sml_lib_project/sml_lib.csv",    # sml_lib.csv 경로 지정
    "file2": "data/sml_lib_project/book_hl.csv",    # book_hl.csv 경로 지정
    "file3": "data/sml_lib_project/book_list.csv",  # book_list.csv 경로 지정
}

# JSON 파일 불러오기 + 변환 함수
def process_file(json_file, js_count):
    file_path = os.path.join(json_dir, json_file)

    # JSON 파일 열기 : 닉네임 때문에 UTF-8로 인코딩해야함
    with open(file_path, 'r', encoding='utf-8') as f:
        data = json.load(f)

    match_id = data['matchId']
    match_date = data['matchDate']
```

```python
for info in data['matchInfo']:
    ouid = info['ouid']
    nickname = info['nickname']

    # 작은 도서관 데이터.csv
    match_detail = info.get('sml_lib', {})
    match_detail.update({"matchID": match_id, "matchDate": match_date, "ouid": ouid, "nickname": nickname})
    if os.path.exists(output_files["file1"]):
        pd.DataFrame([match_detail]).to_csv(output_files["file1"], mode='a', index=False, header= False)
    else:
        pd.DataFrame([match_detail]).to_csv(output_files["file1"], mode='w', index=False, header= True)

    # 도서별 반납기.csv
    shoot = info.get('book_hl', {})
    shoot.update({"matchID": match_id, "matchDate": match_date, "ouid": ouid, "nickname": nickname})
    if os.path.exists(output_files["file2"]):
        pd.DataFrame([shoot]).to_csv(output_files["file2"], mode='a', index=False, header= False)
    else:
        pd.DataFrame([shoot]).to_csv(output_files["file2"], mode='w', index=False, header= True)

    # 국중 장서목록.csv
    passes = info.get('book_list', {})
    passes.update({"matchID": match_id, "matchDate": match_date, "ouid": ouid, "nickname": nickname})
    if os.path.exists(output_files["file3"]):
        pd.DataFrame([passes]).to_csv(output_files["file3"], mode='a', index=False, header= False)
    else:
        pd.DataFrame([passes]).to_csv(output_files["file3"], mode='w', index=False, header= True)


# 모든 JSON 파일 처리 함수
def process_all_files():
    json_files = [f for f in os.listdir(json_dir) if f.endswith('.json')]

    for js_count, json_file in enumerate(json_files, start=1):
        process_file(json_file, js_count)
        print(f"{json_file} 변환 완료 | {js_count}")

    print(f"총 {len(json_files)}개의 JSON 파일 변환 완료!")
```

데이터 수집, 전처리

도서관 정보나루에서 제공중인 OPEN API와, 문화 포털에 국립중앙도서관이 업로드한 **데이터셋을 수집**하여 기존 JSON / XML 형식의 데이터 파일을 분석에 용이하도록 CSV 형태로 변환하였습니다.

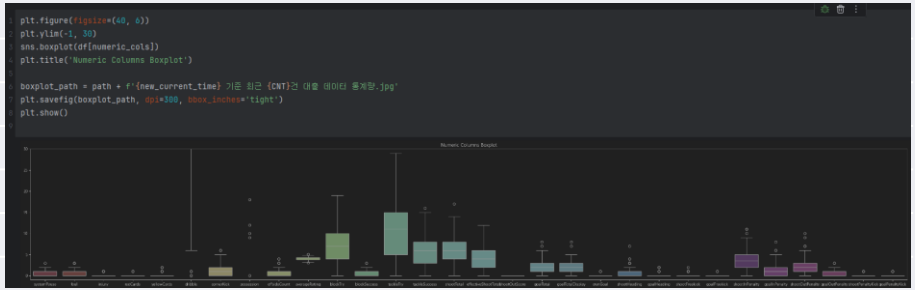
또한, 수집한 데이터에서 **결측치**가 있다면, 해당 행을 제거하고 **Z-score**를 이용하여 **이상치를 판별**해 제거했습니다. **(신뢰구간 95%)**

프로젝트 초기 진행상황 – EDA

```python
def create_pitch():
    # 배경 색상
    fig, ax = plt.subplots(figsize=(12, 7))
    ax.set_facecolor('green')

    # 경계선
    ax.plot([0, pitch_length], [0, 0], color='white', lw=3)
    ax.plot([pitch_length, pitch_length], [0, pitch_width], color='white', lw=3)
    ax.plot([pitch_length, 0], [pitch_width, pitch_width], color='white', lw=3)
    ax.plot([0, 0], [pitch_width, 0], color='white', lw=3)

    # 평균값
    ax.plot([pitch_length / 2, pitch_length / 2], [0, pitch_width], color='white', lw=3)

    # 중앙값
    ax.plot([0, 0], [pitch_width / 2 + 9, pitch_width / 2 - 9], color='white', lw=3)
    ax.plot([pitch_length, pitch_length], [pitch_width / 2 + 9, pitch_width / 2 - 9], color='white', lw=3)

    # 박스플롯
    center_circle = plt.Circle((pitch_length / 2, pitch_width / 2), 9.15, color='white', fill=False, lw=3)
    ax.add_patch(center_circle)

    # 분포
    left_penalty_spot = plt.Circle((11, pitch_width / 2), 0.2, color='white')
    right_penalty_spot = plt.Circle((pitch_length - 11, pitch_width / 2), 0.2, color='white')
    ax.add_patch(left_penalty_spot)
    ax.add_patch(right_penalty_spot)

    # 경계 설정
    ax.set_xlim(0, pitch_length)
    ax.set_ylim(0, pitch_width)

    return ax

# 배경 그리기
ax = create_pitch()

# 제목 추가
plt.title('대출 데이터 통계량')
```

EDA

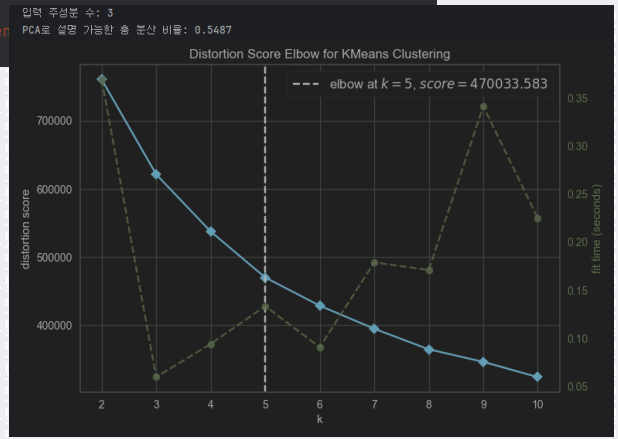
수집된 데이터의 **통계적 수치 확인**과, 각 컬럼간의 **상관관계 등을 파악**하여 본격적인 데이터 분석에 앞서 **데이터의 성질과, 그 특징을 파악**했습니다.



프로젝트 초기 진행상황 – 머신러닝

```python
# 주성분 수를 정했다면 엘보우 그래프 통해 K-means의 적절한 클러스터 수 결정
def check_cluster(group=g0, pca_components=3, check_range=(2,11)):
    # pca 실행
    pca = PCA(n_components=pca_components)
    pca_df = pca.fit_transform(df[group])

    print(f"입력 주성분 수: {pca_components}")
    print(f"PCA로 설명 가능한 총 분산 비율: {pca.explained_variance_ratio_.sum():.4f}")

    # 엘보우 그래프 시각화
    model = KMeans()
    visualizer = KElbowVisualizer(model, k=check_range)
    visualizer.fit(pca_df)
    visualizer.show()

### 함수 실행
check_cluster(group=g3, pca_componer
[9]
```

```
입력 주성분 수: 3
PCA로 설명 가능한 총 분산 비율: 0.5487
```



ML : 군집분석, 회귀분석

작은 도서관의 대출패턴을 파악하기 위해 우선적으로 **도서들을 그룹화** 하여 그 대출 패턴을 파악하는 **군집분석을 시행**했습니다.

이후에는 **어떤 도서가 얼마나 대출될지 예측**하는 **회귀분석을 진행**했습니다.

프로젝트 초기 진행상황 – 머신러닝 고도화

```python
from sklearn.metrics import confusion_matrix, classification_report
from sklearn.model_selection import train_test_split

# 1. 특징 변수(X)와 타겟 변수(y) 정의
X = df.drop(columns=['ouid', 'controller', 'matchResult', 'matchID'])  # 불필요한 열 제거
y = df['matchResult']

# 2. 데이터 분할
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42, stratify=y)

# 3. 최적 하이퍼파라미터로 모델 학습
best_rf_model.fit(X_train, y_train)

# 4. 테스트 데이터 예측
y_pred = best_rf_model.predict(X_test)

# 5. 모델 평가
conf_matrix = confusion_matrix(y_test, y_pred)
class_report = classification_report(y_test, y_pred, target_names=['패', '승'])

# 6. 출력
print("Confusion Matrix:\n", conf_matrix)
print("\nClassification Report:\n", class_report)
```

```
Confusion Matrix:
 [[43762 22191]
 [21859 44094]]

Classification Report:
              precision    recall  f1-score   support

           패       0.67      0.66      0.67     65953
           승       0.67      0.67      0.67     65953

    accuracy                           0.67    131906
   macro avg       0.67      0.67      0.67    131906
weighted avg       0.67      0.67      0.67    131906
```

머신러닝 고도화

AutoGluon, Optuna 등 머신러닝 모델의 품질을 상승시킬 수 있는 도구를 **활용하여 ML 모델의 고도화**에 시간을 투자했습니다.

하지만, **최종 결과 재현율 33%**로 기대에 미치지 못하는 결과물이 되었습니다.

프로젝트 초기 진행상황 – 메일링 시스템

```python
import os
import requests
from slack_sdk import WebClient
from slack_sdk.errors import SlackApiError
from email.mime.text import MIMEText
from email.mime.multipart import MIMEMultipart
from email.mime.image import MIMEImage
from email.mime.base import MIMEBase
from email import encoders

# 파일 첨부 포함 이메일 생성 함수
def make_multi_msg(content, file_paths=None):
    multi = MIMEMultipart(_subtype="mixed")  # MIMEMultipart 객체 생성
    multi.attach(MIMEText(content, 'html', _charset="utf-8"))  # 이메일 본문 추가

    if inline_images:
        for idx, img_path in enumerate(inline_images):
            with open(img_path, "rb") as file:
                img = MIMEImage(file.read())
                img.add_header("Content-ID", f"<image{idx}>")  # cid
                img.add_header("Content-Disposition", "inline", filen
                multi.attach(img)

    # 파일 첨부가 있을 경우
    if file_paths:
        for file_path in file_paths:
            with open(file_path, "rb") as file:
                msg = MIMEBase("application", "octet-stream")
                msg.set_payload(file.read())
                encoders.encode_base64(msg)

                filename = os.path.basename(file_path)
                msg.add_header("Content-Disposition", "attachment", f
                multi.attach(msg)  # 첨부파일 추가

    return multi  #MIMEMultipart 객체 반환
```

```python
# Slack 메시지 전송 (Webhook)
def send_slack_message(message):
    headers = {"Content-Type": "application/json"}
    data = {"text": message}
    res = requests.post(SLACK_WEBHOOK_URL, headers=headers, json=data)

    if res.status_code == 200:
        print("✅ Slack 메시지 전송 성공")
    else:
        print(f"❌ Slack 메시지 전송 실패: {res.status_code}")

# Slack 파일 업로드 (`files_upload_v2` 사용)
def upload_slack_files(file_paths, message="📄 분석 결과 파일입니다."):
    try:
        for file_path in file_paths:
            file_ext = file_path.split(".")[-1].lower()
            filetype = "csv" if file_ext == "csv" else "auto"  # CSV 파일일 경우 `filetype="csv"`

            with open(file_path, "rb") as file:
                response = slack_client.files_upload_v2(
                    channel=SLACK_CHANNEL_ID,
                    initial_comment=message,
                    file=file,
                    filename=file_path.split("/")[-1],  # 파일 이름 지정
                    filetype=filetype,  # CSV 파일은 `csv`로 지정
                )

            print(f"✅ {file_path} 파일 업로드 성공!")

    except SlackApiError as e:
        print(f"❌ Slack 파일 업로드 실패: {e.response['error']}")

# SMTP 서버 정보
SMTP_INFO = {
    "smtp_server": "smtp.gmail.com",
    "smtp_user_id": my_email,
    "smtp_user_pw": my_password,
    "smtp_port": 587                    # 구글 SMTP 포트(고정)
}
```

메일링 시스템 구축

앞서 제작한 머신러닝 모델을 바탕으로 매일 / 매주 / 매월 / 매 분기 데이터를 취합하여 **주기적으로 이메일로 발송**하거나, Slack과 같은 **업무용 메신저로 자동으로 데이터 분석 리포트를 발송**하는 **메일링 시스템을 구축**했습니다.

피보팅

분석된 결과의 **메일링 시스템을 구축**하고, **실시간 반영 대시보드**를 제작하면 **겉으로 보기에는 그럴듯한 프로젝트가** 되지만, 이렇게 만들어진 결과물이 **절대 유의미한 인사이트를 만들어낼 수 없다**고 생각했습니다.

- ➔ 수집한 **데이터의 범위**가 충분하지 못했고,
- ➔ **ML 모델의 정확도**가 낮았으며,
- ➔ 그로 인해 **결과물도 만족스럽지 못했습니다.**

상기한 이유들로 인해, 이대로 프로젝트를 마치는 것 보다, **현재 상황을 해결할 방안을 찾는 것이 더 유익**하리라 생각했습니다.

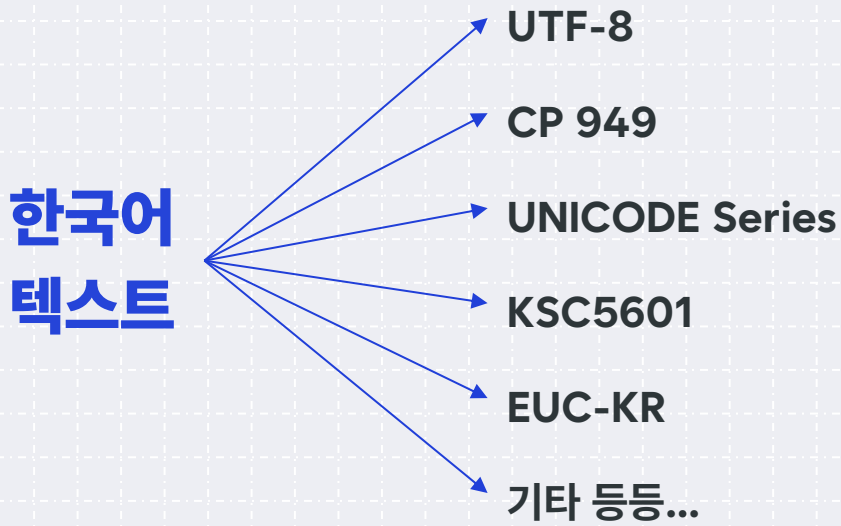
프로젝트 기간이 2주 남짓 남은 촉박한 시점이었지만, 기존에 프로젝트를 진행하며 **도서관 데이터 공유 현황**과, 해당 **데이터를 활용하여 분석**하는 **검증 과정을 거쳤기 때문에**, 시간이 충분하리라 예상했습니다.

02

데이터 공유 현황의 주요 문제점

데이터 분석 과정에서 조우한 현행 데이터 공유 시스템의 주요 문제점에 대해 설명합니다.

문제점 1 : 인코딩 포맷



인코딩 포맷

같은 기관에서 비슷한 날짜에 배포한 데이터의 경우에도, **서로 인코딩 포맷이 달라** 데이터셋을 정상적으로 활용하기 위해선 **14종류 이상의 한국어 인코딩 포맷** 중, 어떤 포맷을 사용했는지 **하나씩 시도해 보아야 하는 불편함**이 있습니다.

문제점 2 : 데이터 파편화

데이터 파편화

도서의 서지정보를 얻고 싶다면, **서로 다른 사이트에서 제공중인 데이터셋을 서로 병합**하여 사용해야 하고, 그 마저도 **서로 공유 시점이 달라 데이터의 신뢰성을 확신할 수 없는 상황**입니다.

03

프로젝트 진행 과정

앞서 설명 드린 현행 데이터 공유 시스템의 문제점의 해결법을 제안하기 위한 프로젝트 수행 과정에 대해 설명합니다.

프로젝트 수행 과정


공유 데이터를 바탕으로 데이터 분석
데이터 공유 시스템의 미흡한 점 발견
데이터 공유 시스템 개선점 분석
데이터 공유 시스템 개선점 정리 및 해결방안 제안
노션 페이지에 정리
기존 데이터 공유 우수사례 조사
데이터 공유 우수사례 큐레이션
데이터 공유 우수사례를 활용한 데이터 활용 사례 큐레이션

04

추후 프로젝트 개선 사항

프로젝트 결과물을 시연하고, 추후 개선방안에 대해 설명합니다.

노션 페이지로 이동

참고문헌

- Jake VanderPlas. **『파이썬 데이터 사이언스 핸드북』.** 위키북스, 2023.
- 인하대학교 통계학과. **『통계학』.** 자유아카데미, 2022.
- Peter Bruce. **『Practical Statistics for Data Scientists, 2nd Edition』.** O`Reilly, 2020.
- Wes McKinney. **『Python for Data Analysis, 3rd Edition』.** O`Reilly, 2022.
- Ville Tuulos. **『Effective Data Science Infrastructure』.** Manning Publications, 2022.
- Mona Khalil. **『Effective Data Analysis』.** Manning Publications, 2025.
- **문화 빅데이터 포털**
- **도서관 정보나루**
- 국립중앙도서관. (2024). **2024년 도서관 데이터 분석 활용 사례집.** 국립중앙도서관