소상공인시장진흥공단 상가업소정보로

스타벅스, 이디야 위치 분석하기

시스템제어공학과

20210710 맹지우

Content

Introduction

서론_배경 및 목적

Main Body

본론_코드 구현

Conclusion

결론_결과 분석

01

서론

Introduction

배경 및 목적

소상공인시장진흥공단이 제공하는 "상가업소정보" 데이터는 전국의 상권 정보를 담고 있습니다. 이 데이터를 활용하여 특정 지역의 상권을 분석하고, 스타벅스와 이디야 카페의 위치를 파악하는 것은 상권 조사와 카페 시장 동향을 이해하는 데에 도움이 됩니다.

스타벅스와 이디야는 대표적인 카페 브랜드로서 경쟁 관계에 있습니다. 이 두 브랜드의 위치와 분포를 비교하여, 분석 결과를 시각화 하여 쉽게 이해할 수 있도록 코드를 작성하고자 합니다.

Algorithm

02

코드 구현

모듈 불러오기 & 한글 폰트 설정

```
1  import pandas as pd
2  import numpy as np
3  import seaborn as sns
```

matplotlib의 pyplot과 matplotlib의 font_manager 폰트 관련 기능을 제공하는 모듈 불러오기

```
1  # 한글폰트 설정
2  import matplotlib.pyplot as plt
3  import matplotlib.font_manager as fm
4
5  font_location = 'C:₩₩WINDOWS₩₩Fonts₩₩HancomHoonminjeongeumH.ttf' # For Windows
6  font_name = fm.FontProperties(fname=font_location).get_name()
7  matplotlib.rc('font', family=font_name)
```

matplotlib의 pyplot과 matplotlib의 font_manager 폰트 관련 기능을 제공하는 모듈 불러오기

5번째 줄

사용할 한글 폰트 파일의 경로를 지정

6번째 줄

fm.FontProperties()를 사용하여 폰트 파일을 지정하고, get_name() 함수를 호출하여 폰트의 이름 가져오기

7번째 줄

Plt.rc()를 사용하여 전역적으로 폰트 패밀리 설정

위와 같은 코드를 실행하면 matplotlib에서 사용하는 폰트 패밀리가 설정된 한글 폰트로 변경되어, 이후에 생성하는 그래프나 텍스트 요소들은 설정한 한글 폰트를 사용하여 표시됩니다.

데이터프레임 변환

```
df = pd.read_csv("상가업소정보.csv", sep='|')
df.shape
```

```
(573680, 39)
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 573680 entries, 0 to 573679
Data columns (total 39 columns):
 #   Column      Non-Null Count   Dtype
---  ------      --------------   -----
 0   상가업소번호      573680 non-null  int64
 1   상호명         573679 non-null  object
 2   지점명         76674 non-null   object
 3   상권업종대분류코드   573680 non-null  object
 4   상권업종대분류명    573680 non-null  object
 5   상권업종중분류코드   573680 non-null  object
 6   상권업종중분류명    573680 non-null  object
 7   상권업종소분류코드   573680 non-null  object
 8   상권업종소분류명    573680 non-null  object
 9   표준산업분류코드    539290 non-null  object
 10  표준산업분류명     539290 non-null  object
 11  시도코드        573680 non-null  int64
 12  시도명         573680 non-null  object
 13  시군구코드       573680 non-null  int64
 14  시군구명        573680 non-null  object
 15  행정동코드       573680 non-null  int64
 16  행정동명        573680 non-null  object
 17  법정동코드       573680 non-null  int64
 18  법정동명        573680 non-null  object
 19  지번코드        573680 non-null  int64
 20  대지구분코드      573680 non-null  int64
 21  대지구분명       573680 non-null  object
 22  지번본번지       573680 non-null  int64
 23  지번부번지       474924 non-null  float64
 24  지번주소        573680 non-null  object
```

pd.read_csv("상가업소정보.csv", sep='|')

"상가업소정보.csv" 파일을 파이썬의 pandas 라이브러리를 사용하여 데이터프레임으로 읽어옵니다. sep='|'는 데이터의 구분자가 파이프(|)로 되어 있음을 지정해줍니다.

df.shape

읽어온 데이터프레임의 행과 열의 개수를 출력합니다.

df.info()

데이터프레임의 각 열에 대한 정보를 출력합니다. 열의 데이터 타입과 누락된 값의 개수 등을 확인할 수 있습니다.

사용할 컬럼 분류

```python
columns = ['상호명', '상권업종대분류명', '상권업종중분류명', '상권업종소분류명',
           '시도명', '시군구명', '행정동명', '법정동명', '도로명주소',
           '경도', '위도']

df = df[columns].copy()
df.shape
```

```
(573680, 11)
```

```python
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 573680 entries, 0 to 573679
Data columns (total 11 columns):
 #   Column     Non-Null Count   Dtype
---  ------     --------------   -----
 0   상호명        573679 non-null  object
 1   상권업종대분류명  573680 non-null  object
 2   상권업종중분류명  573680 non-null  object
 3   상권업종소분류명  573680 non-null  object
 4   시도명        573680 non-null  object
 5   시군구명       573680 non-null  object
 6   행정동명       573680 non-null  object
 7   법정동명       573680 non-null  object
 8   도로명주소      573680 non-null  object
 9   경도         573680 non-null  float64
 10  위도         573680 non-null  float64
dtypes: float64(2), object(9)
memory usage: 48.1+ MB
```

선택할 컬럼들의 이름을 담은 리스트입니다.

df[columns].copy()

원본 데이터프레임에서 columns 리스트에 포함된 컬럼들만 선택하여 새로운 데이터프레임을 만듭니다. copy() 메서드를 사용하여 복사본을 생성합니다.

df.shape

선택된 컬럼들로 구성된 새로운 데이터프레임의 행과 열의 개수를 출력합니다.

df.info()

새로운 데이터프레임의 각 열에 대한 정보를 출력합니다. 열의 데이터 타입과 누락된 값의 개수 등을 확인할 수 있습니다.

데이터 변수 저장

```
df_seoul = df[df["시도명"] == "서울특별시"].copy()
df_seoul.shape
```

```
(407376, 11)
```

"시도명" 열이 "서울특별시"인 조건을 만족하는 데이터만 선택하여 복사합니다.
df_seoul 데이터프레임의 행과 열의 개수를 출력합니다.

```
# 문자열 소문자로 변경
df_seoul["상호명_소문자"] = df_seoul["상호명"].str.lower()
```

df_seoul 데이터프레임의 "상호명" 열의 문자열을 소문자로 변환하여 "상호명_소문자" 열에 저장합니다.

```
df_seoul.loc[df_seoul["상호명_소문자"].str.contains("이디야|이디아|ediya"), "상호명_소문자"].shape
```

```
(543,)
```

"상호명_소문자" 열에서 "이디야", "이디아", "ediya"와 같은 문자열을 포함하는 행만 선택하여 해당 행들의 개수를 출력합니다.

```
df_seoul.loc[df_seoul["상호명_소문자"].str.contains("스타벅스|starbucks"), "상호명_소문자"].shape
```

```
(506,)
```

"상호명_소문자" 열에서 "스타벅스", "starbucks"와 같은 문자열을 포함하는 행만 선택하여 해당 행들의 개수를 출력합니다.

```
df_cafe = df_seoul[df_seoul["상호명_소문자"].str.contains('스타벅스|starbucks|이디야|이디아|ediya')].copy()
df_cafe.shape
```

```
(1049, 12)
```

"상호명_소문자" 열에서 "스타벅스", "starbucks", "이디야", "이디아", "ediya"와 같은 문자열을 포함하는 행만 선택하여 새로운 데이터프레임 df_cafe를 생성하고, 해당 데이터프레임의 행과 열의 개수를 출력합니다.

데이터 변수 저장

```
df_cafe.loc[df_cafe["상호명_소문자"].str.contains('스타벅스|starbucks'), "브랜드명"] = "스타벅스"
df_cafe.loc[~df_cafe["상호명_소문자"].str.contains('스타벅스|starbucks'), "브랜드명"] = "이디야"
df_cafe[["상호명_소문자", "브랜드명"]].head()
```

	상호명_소문자	브랜드명
1104	스타벅스	스타벅스
1675	이디야커피	이디야
2023	스타벅스종로3가점	스타벅스
2770	스타벅스	스타벅스
2957	이디야커피	이디야

"상호명_소문자" 열에서 "스타벅스", "starbucks"와 관련된 행의 "브랜드명"을 "스타벅스"로 설정하고, 그 외의 행의 "브랜드명"을 "이디야"로 설정합니다. 그리고 "상호명_소문자"와 "브랜드명" 열만 선택하여 처음 5개의 행을 출력하여 확인합니다.

```
# df_cafe에 담긴 '상호명', '브랜드명'으로 미리보기를 합니다.

df_cafe[["상호명", "브랜드명"]].tail()
```

	상호명	브랜드명
567090	스타벅스	스타벅스
567828	스타벅스	스타벅스
568636	이디야커피	이디야
570096	스타벅스	스타벅스
571052	스타벅스	스타벅스

"df_cafe" 데이터프레임에서 "상호명"과 "브랜드명" 열만 선택하여 마지막 5개의 행을 미리보기로 출력합니다.

스타벅스와 이디야의 점포 개수 시각적 비교

```
1 df_cafe["브랜드명"].value_counts()
```

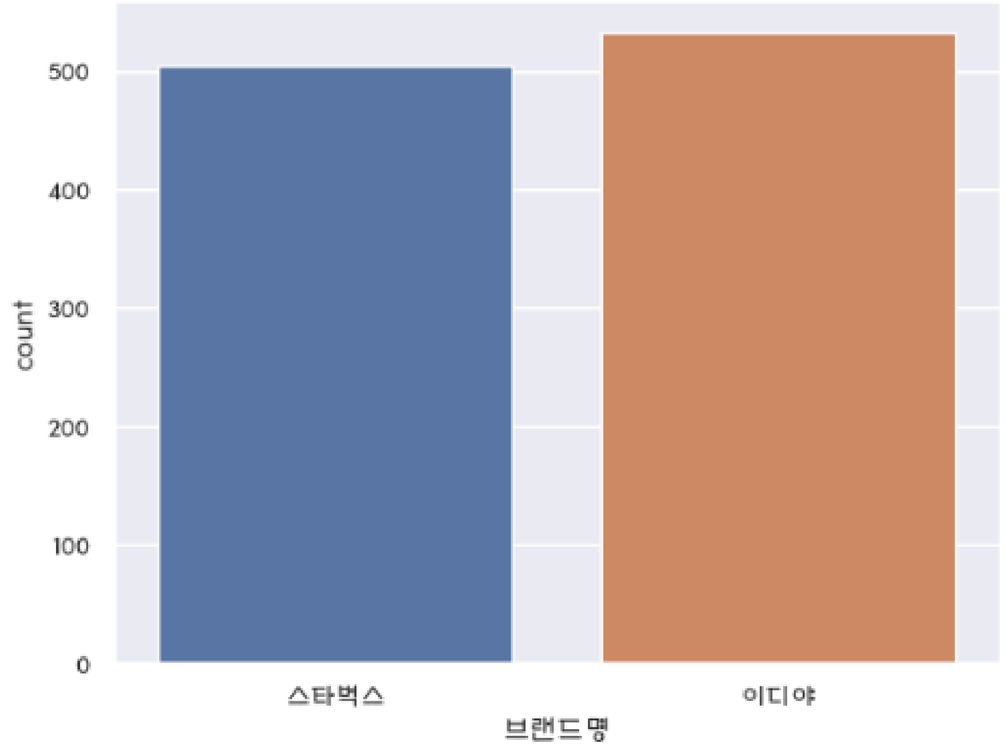
```
이디야     532
스타벅스    504
Name: 브랜드명, dtype: int64
```

"df_cafe" 데이터프레임의 "브랜드명" 열에 있는 값들의 개수를 세어줍니다

```
1 sns.countplot(data=df_cafe, x="브랜드명")
```

```
<Axes: xlabel='브랜드명', ylabel='count'>
```

Seaborn 라이브러리의 countplot을 사용하여 "df_cafe" 데이터프레임에서 "브랜드명" 열의 값들의 개수를 시각화하는 역할을 합니다.

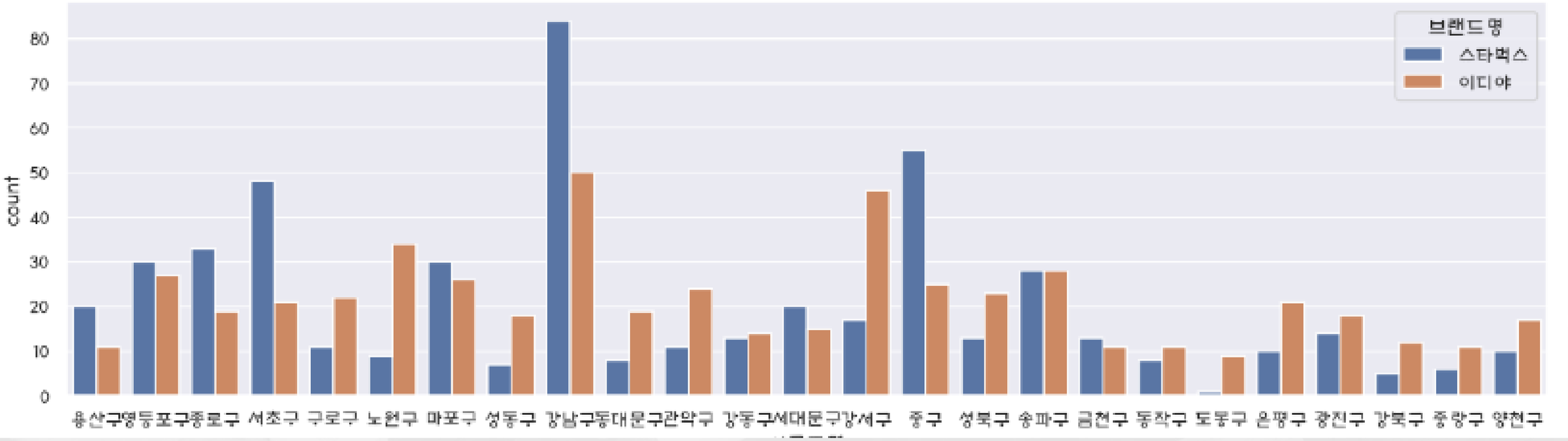


스타벅스와 이디야의 시군구에 따른 점포 개수 시각적 비교

Matplotlib의 figure와 Seaborn의 countplot을 함께 사용하여 "df_cafe" 데이터프레임에서 "시군구명" 열의 값들의 개수를 시각화하고, "브랜드명"에 따라 색상을 구분하여 표현해줍니다.

```
1 plt.figure(figsize=(15, 4))
2 sns.countplot(data=df_cafe, x="시군구명", hue="브랜드명")
```

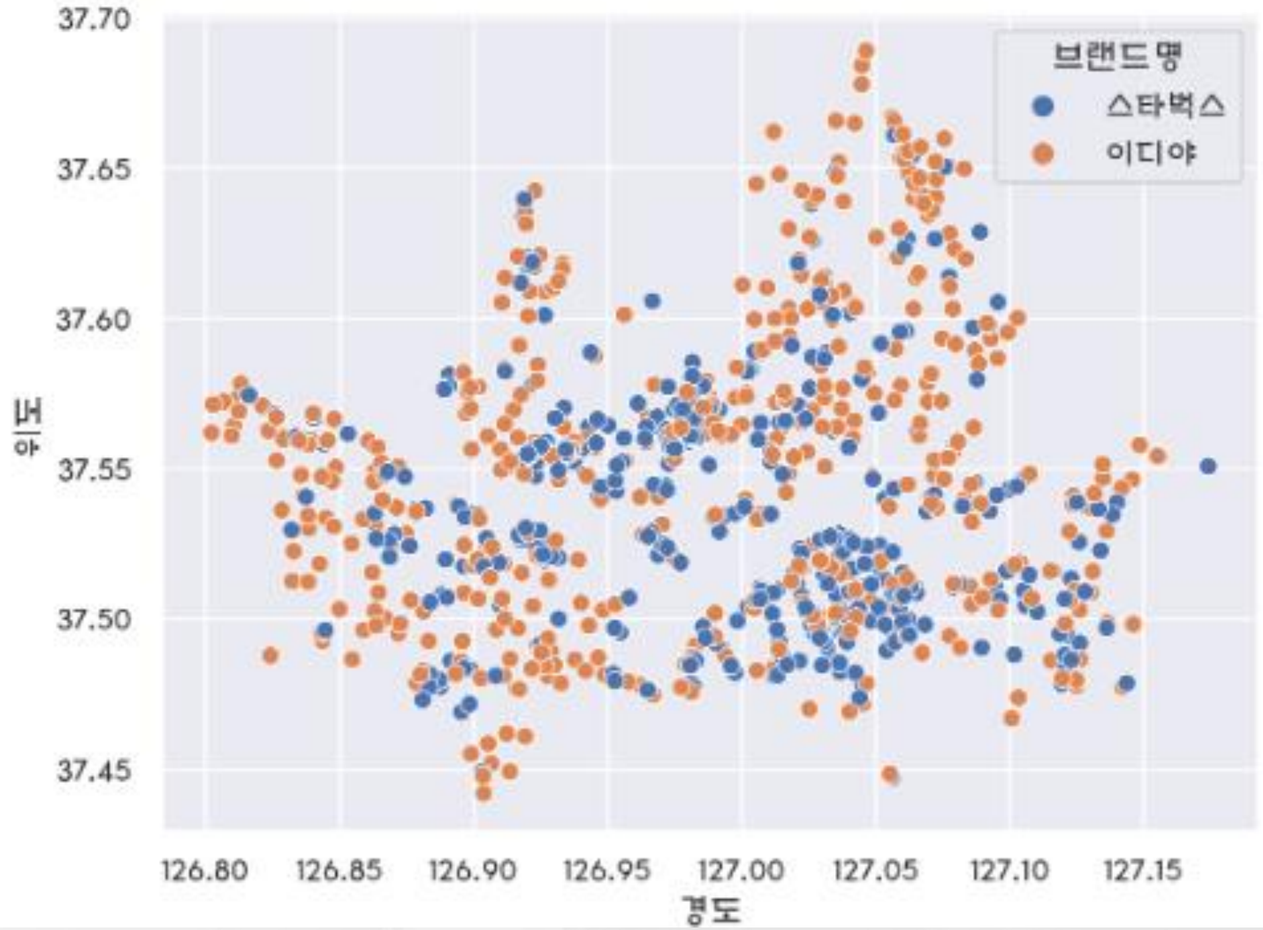
```
<Axes: xlabel='시군구명', ylabel='count'>
```



스타벅스와 이디야의 시군구에 따른 점포 개수 시각적 비교

```
1 sns.scatterplot(data=df_cafe, x="경도", y="위도", hue="브랜드명")
```

```
<Axes: xlabel='경도', ylabel='위도'>
```



Seaborn의 scatterplot을 사용하여 "df_cafe" 데이터프레임의 위도(위도)와 경도(경도) 정보를 산점도로 시각화하고, 각 점을 "브랜드명"에 따라 색상으로 구분합니다.

03

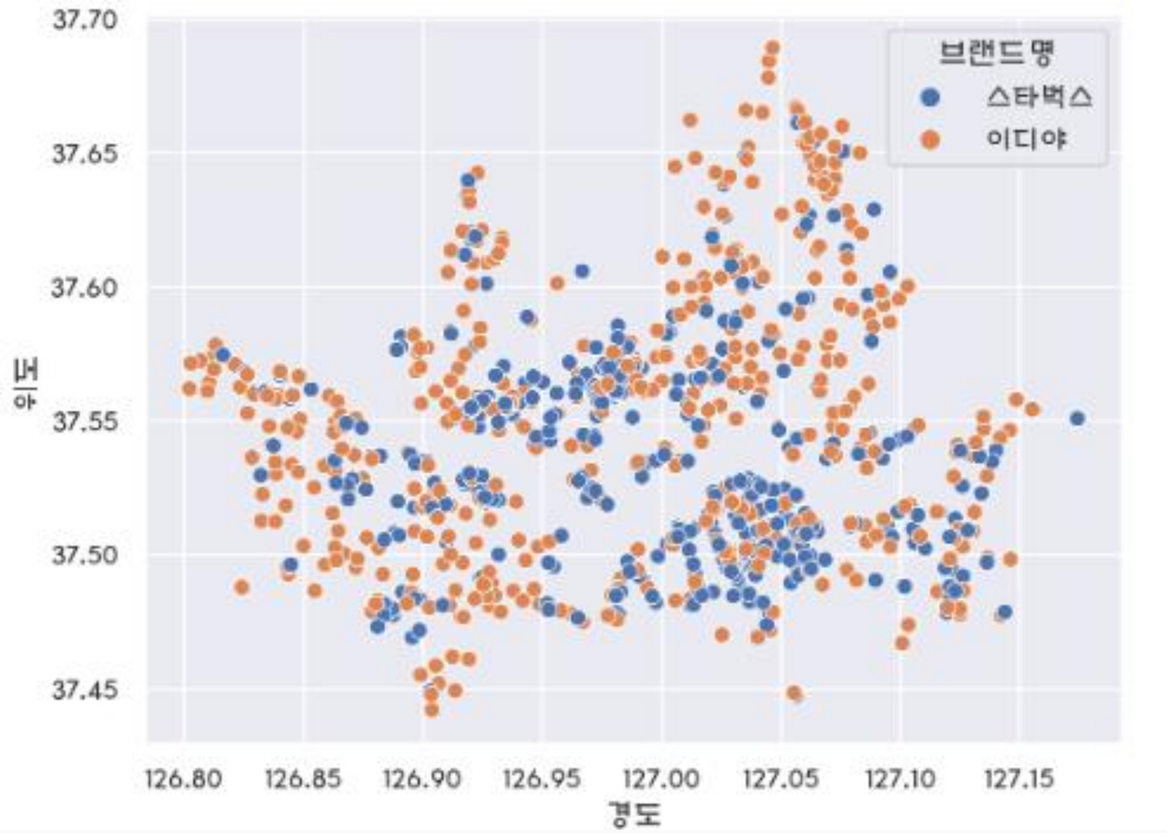
결론

Conclusion

결론

"상가업소정보" 데이터를 활용하여 서울특별시 내의 스타벅스와 이디야 카페의 위치를 분석해보았습니다. 데이터 수집에서는 소상공인시장진흥공단에서 제공하는 데이터로부터 수집하였으며, 데이터 전처리를 통해 필요한 열만 선택하고, 서울특별시에 해당하는 데이터만 추출하였습니다. 또한, 상호명을 소문자로 변환하여 검색을 용이하게 하였습니다. 그리고 상호명을 기준으로 스타벅스와 이디야 카페를 구분하였습니다. 데이터를 가공한 후, 보기 쉽도록 Seaborn 라이브러리를 사용하여 브랜드별 분포를 시각화 하여, "브랜드명"에 따라 막대 그래프와 산점도를 그려 브랜드별로 위치와 개수를 한눈에 확인할 수 있었습니다.

결론



위도와 경도를 통해 산점도로 카페의 분포를 살펴본 결과, 서울특별시의 상권에서 스타벅스와 이디야 카페는 많이 분포하였습니다. 이 위치를 살펴보면 스타벅스는 주로 상업 지역이나 번화가에 위치하고 있었던 반면에, 이디야는 번화가가 아니더라도 더 다양한 지역에 분포해 있음을 볼 수 있었습니다.

THANK YOU