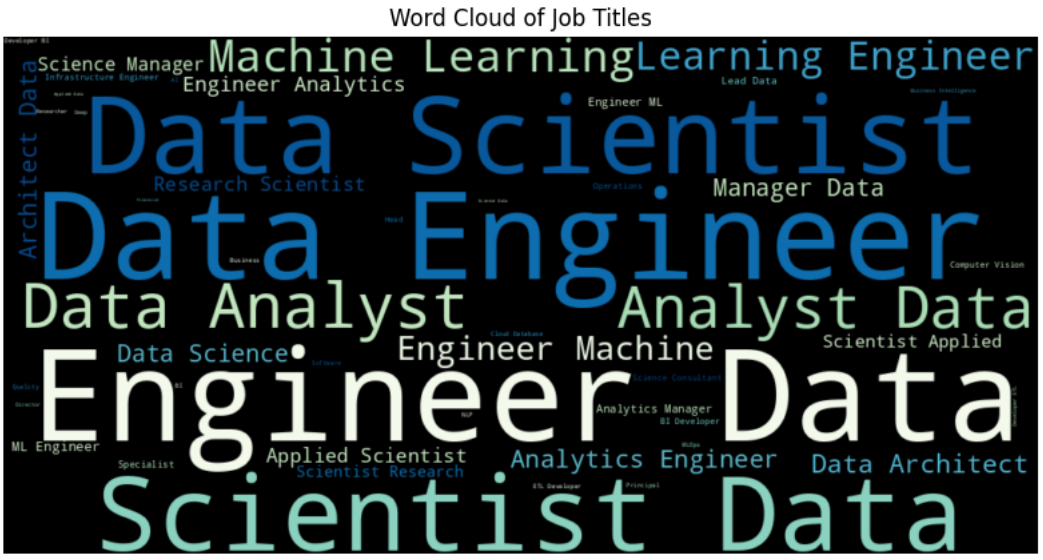
7조

팀장 | 소연 황



Frequent mentioned Job Titles related to Data Science (Word Cloud)

데이터톤 세부 추진

Aa Name	Tags	Date
주제 아이데이션	팀원2	@2023년 9월 6일 → 2023년 9월 9일
자료조사	팀장	@2023년 9월 6일 → 2023년 9월 9일
데이터 수집	팀원2	@2023년 9월 6일 → 2023년 9월 9일
데이터 분석 / EDA / Machine Learning	팀원2	@2023년 9월 8일 → 2023년 9월 19일
중간구현 기획서 제출		@2023년 9월 8일 → 2023년 9월 14일
분석보고서 작성	팀원	@2023년 9월 11일 → 2023년 9월 21일
발표자료 작성	팀원2	@2023년 9월 13일 → 2023년 9월 21일

1. Data Introduction

Data Science Salary 2021 to 2023 : from Kaggle

- **Data Science Salary 💰 2021 to 2023**
  https://www.kaggle.com/datasets/harishkumardatalab/data-science-salary-2021-to-2023

Columns

- work_year
- experience_level

EN (Entry-Level), EX (Experienced), MI (Mid-Level), SE (Senior).

- employement_type

FT (Full-Time), CT (Contractor), FL (Freelancer), PT (Part-Time).

- job_title
- salary
- salary_currency
- salaray_in_usd
- company_location
- company_size

2. Data Cleaning

```
df = pd.read_csv('Data Science Salary 2021 to 2023.csv')
df.isnull().sum() # null값 확인

filt = df['company_location'] == 'US'
# Among total 3761 rows 3045 rows are dealing with US occupations
# 3671 행 중 3045개의 행이 US
# 미국 Data Science Industry Job distribution으로 분석주제 좁힘

df = df[filt]
df = df.reset_index(drop=True)
```

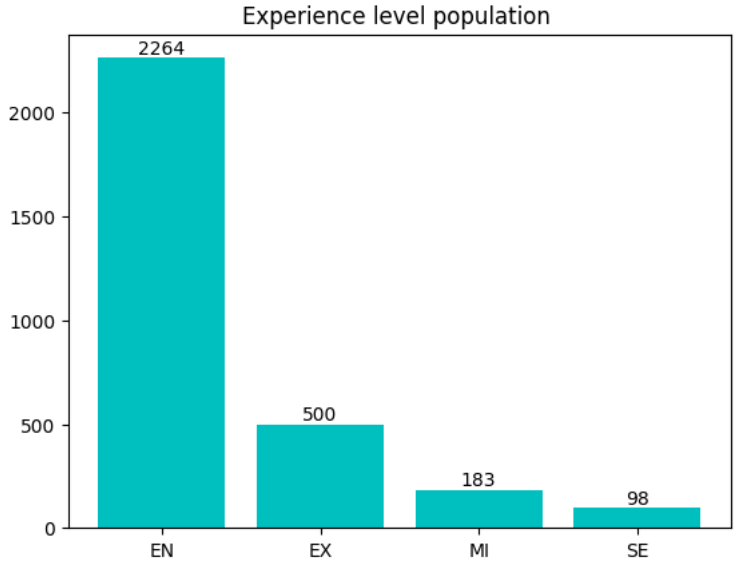
3. EDA

1) Experience Level & Salary

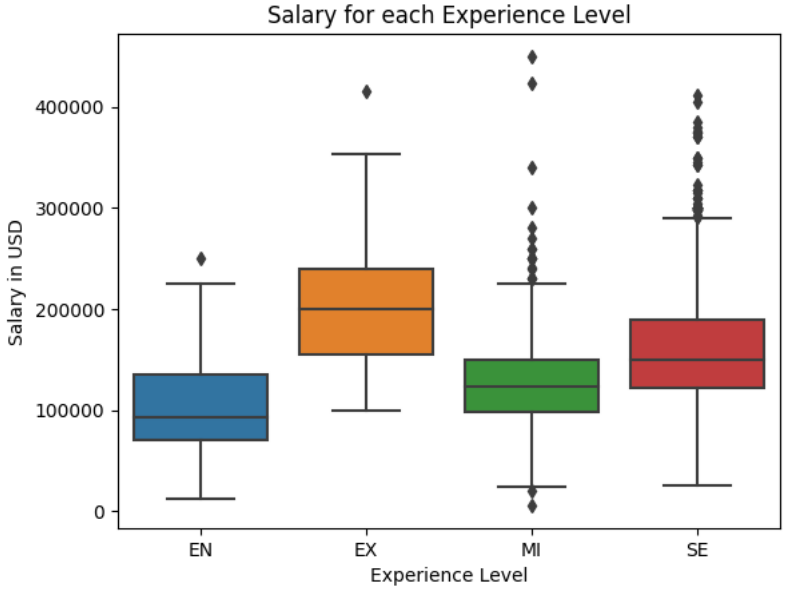
```
df['experience_level'].value_counts()
```

```
# 실행 결과값

SE    2264
MI     500
EN     183
EX      98
```

```
plt.bar(df['experience_level'].unique(),df['experience_level'].value_counts(),color='c')

# 직종 별 인원 수 작성
for index, value in enumerate(df['experience_level'].value_counts()):
    plt.text(index, value, value, ha='center', va='bottom')
plt.title('Experience level population ')

plt.show()
```




```
# Salary for each experience level (2) - boxplot

sns.boxplot(x='experience_level', y='salary_in_usd', data=df)
plt.title('Salary for each Experience Level')
plt.xlabel('Experience Level')
plt.ylabel('Salary in USD')
plt.show()
```




```
# 직급 별 평균 salary
avg = df.groupby("experience_level")["salary_in_usd"].mean()
```

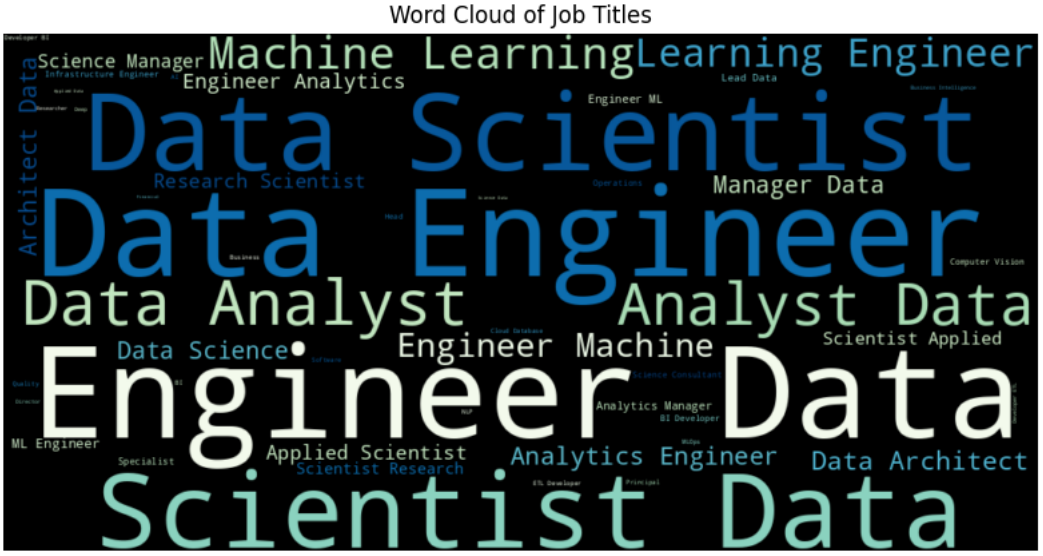
```
# 실행결과
EN    102400.639344
EX    207445.520408
MI    127776.604000
SE    158691.223057
```

```
# Salary for each experience level (2) - avg
```

```
plt.bar(df['experience_level'].unique(),avg,color='c')
plt.title("Average Salary for each experience level")
plt.xlabel("Experience Level")
plt.ylabel("Salary($)")
plt.show()
```

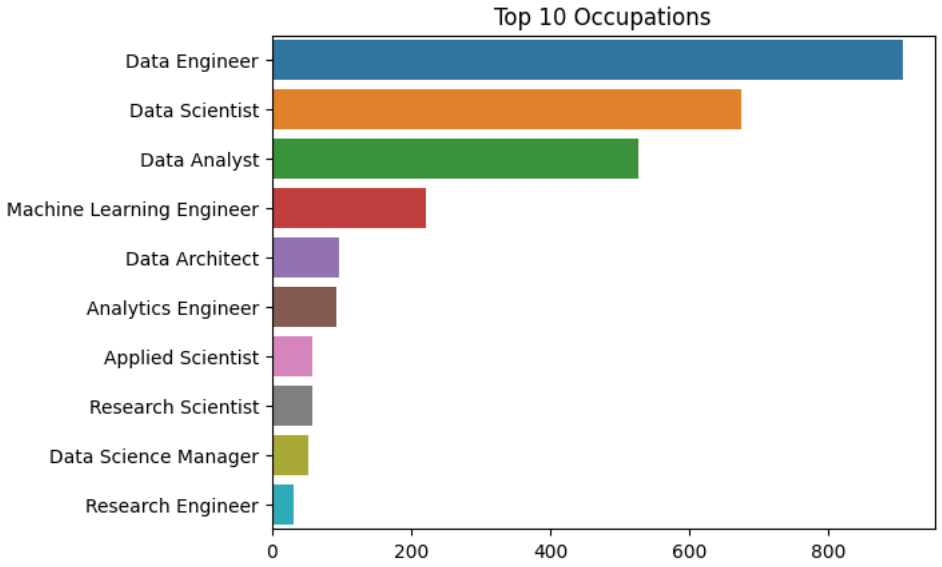
2) Job Title (Frequency)

```
from wordcloud import WordCloud
# Generate a word cloud for job titles
wordcloud = WordCloud(width=1000, height=500, background_color='black',colormap='GnBu', max_words=80).generate(' '.join(df['job_title']

plt.figure(figsize=(10, 6))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis('off')
plt.title('Word Cloud of Job Titles')
# plt.show()
plt.savefig('WordCloudOfJobTitles.pdf')
```

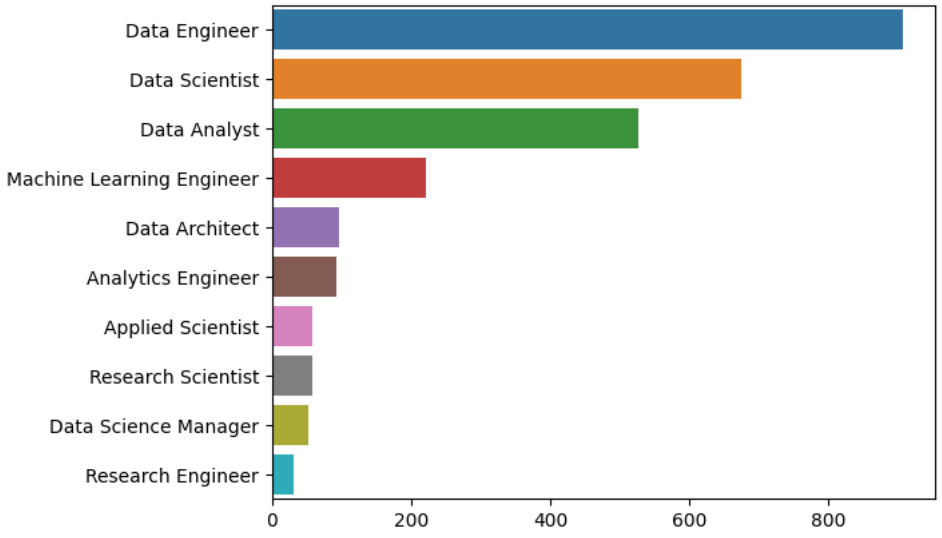



```
mode_job = df.job_title.value_counts()[:10]
sns.barplot(x=mode_job.values, y=mode_job.index, orient="h")
```

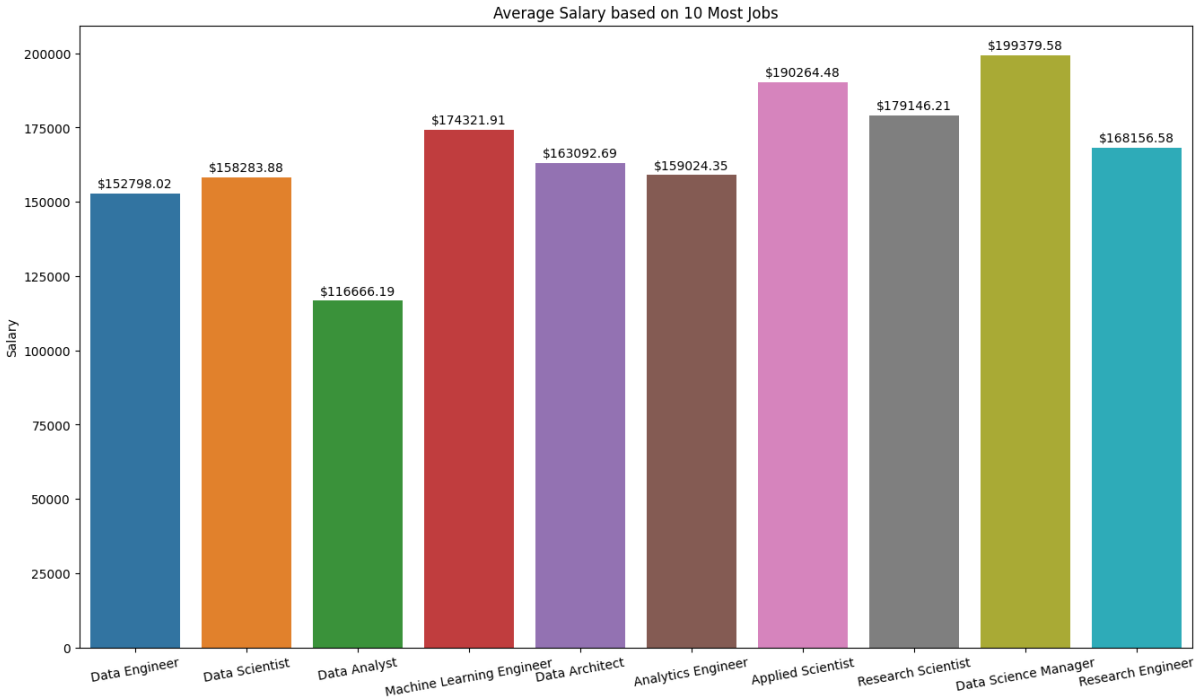
```
plt.title('Top 10 Occupations')
plt.show()
```




```
jobs_top10 = df.job_title.value_counts()[:10]
sns.barplot(x=jobs_top10.values, y=jobs_top10.index, orient="h")
```




```
plt.figure(figsize=(16,9))

jobs_top10_salary = df.groupby("job_title")["salary_in_usd"].mean() \
                    .loc[jobs_top10.index]

ax = sns.barplot(y=jobs_top10_salary.values, x=jobs_top10_salary.index)
for idx, values in enumerate(jobs_top10_salary.values):
    ax.text(idx, values + 1000, f"${values:.2f}", ha='center', va='bottom', fontsize=10)

plt.title("Average Salary based on 10 Most Jobs")
plt.ylabel("Salary")

ax.set_xticklabels(jobs_top10_salary.index, rotation=10)
plt.show()
```



4. Modeling

1) XGBoost

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error, r2_score
import xgboost as xgb

# 변수 범주화
df_one_hot = pd.get_dummies(df, columns=['experience_level', 'employment_type', 'job_title', 'company_location', 'company_size'])

# 불필요한 컬럼 삭제
df_one_hot.drop(['salary_currency'], axis=1, inplace=True)

X = df_one_hot.drop(['salary', 'salary_in_usd'], axis=1)
y = df_one_hot['salary_in_usd']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

xg_reg = xgb.XGBRegressor(objective ='reg:squarederror', colsample_bytree = 0.3, learning_rate = 0.1, alpha = 10, n_estimators = 100)
xg_reg.fit(X_train, y_train)

y_pred = xg_reg.predict(X_test)

rmse = np.sqrt(mean_squared_error(y_test, y_pred))
r2 = r2_score(y_test, y_pred)

print(f"RMSE: {rmse}")
print(f"R2 Score: {r2}")

# 실행결과값
# RMSE: 47160.00192894963
# R2 Score: 0.2590938935430026
```

2) Linear Regression, Decision Tree, Random Forest

```
from sklearn.linear_model import LinearRegression
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score
```

```
# for문 돌리기 위한 모델들 리스트에 넣기
models = [
    ('Linear Regression', LinearRegression()),
    ('Decision Tree', DecisionTreeRegressor(random_state=42)),
    ('Random Forest', RandomForestRegressor(random_state=42))
]

# 각 모델 별 결과 확인하기
for name, model in models:
    model.fit(X_train, y_train)
    y_pred = model.predict(X_test)

    mse = mean_squared_error(y_test, y_pred)
    rmse = mse ** 0.5
    mae = mean_absolute_error(y_test, y_pred)
    r2 = r2_score(y_test, y_pred)

    print(f"Model: {name}")
    print(f"Mean Squared Error: {mse:.2f}")
    print(f"Root Mean Squared Error: {rmse:.2f}")
    print(f"Mean Absolute Error: {mae:.2f}")
    print(f"R-squared: {r2:.2f}")
    print("====================")
```

```
## 실행결과 ##
Model: Linear Regression
Mean Squared Error: 0.97
Root Mean Squared Error: 0.99
Mean Absolute Error: 0.76
R-squared: 0.04
====================
Model: Decision Tree
Mean Squared Error: 0.85
Root Mean Squared Error: 0.92
Mean Absolute Error: 0.72
R-squared: 0.15
====================
Model: Random Forest
Mean Squared Error: 0.86
Root Mean Squared Error: 0.93
Mean Absolute Error: 0.72
R-squared: 0.15
====================
```

5. Conclusion & Insight