```
# import libraries
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
# Suppress all FutureWarnings
warnings.simplefilter(action='ignore', category=FutureWarning)

# Suppress UserWarnings from Seaborn (e.g., for swarmplot)
warnings.simplefilter(action='ignore', category=UserWarning)

file_path = r'C:\Users\Lenovo\Downloads\data_science_salaries.csv'


df = pd.read_csv(file_path)

df.head()
```

```
        job_title experience_level employment_type work_models
work_year  \
0   Data Engineer        Mid-level       Full-time      Remote
2024
1   Data Engineer        Mid-level       Full-time      Remote
2024
2  Data Scientist     Senior-level       Full-time      Remote
2024
3  Data Scientist     Senior-level       Full-time      Remote
2024
4    BI Developer        Mid-level       Full-time     On-site
2024

  employee_residence  salary salary_currency  salary_in_usd
company_location  \
0      United States  148100             USD         148100    United
States
1      United States   98700             USD          98700    United
States
2      United States  140032             USD         140032    United
States
3      United States  100022             USD         100022    United
States
4      United States  120000             USD         120000    United
States

  company_size
0       Medium
1       Medium
2       Medium
```

```
3       Medium
4       Medium

df.tail()

                     job_title experience_level employment_type
work_models  \
6594        Staff Data Analyst      Entry-level        Contract
Hybrid
6595        Staff Data Analyst  Executive-level       Full-time
On-site
6596  Machine Learning Manager     Senior-level       Full-time
Hybrid
6597             Data Engineer        Mid-level       Full-time
Hybrid
6598            Data Scientist     Senior-level       Full-time
On-site

      work_year employee_residence  salary salary_currency
salary_in_usd  \
6594       2020             Canada   60000             CAD
44753
6595       2020            Nigeria   15000             USD
15000
6596       2020             Canada  157000             CAD
117104
6597       2020            Austria   65000             EUR
74130
6598       2020            Austria   80000             EUR
91237

     company_location company_size
6594           Canada        Large
6595           Canada       Medium
6596           Canada        Large
6597          Austria        Large
6598          Austria        Small
```

About the topic

Data science salaries between 2020 and 2024 have seen significant growth due to the increasing demand for data-driven decision-making across industries. As organizations invest more in big data, machine learning, and AI, the need for skilled data scientists has surged. Salaries during this period have been influenced by factors such as location, experience, industry, and specific technical skills like Python, R, SQL, and cloud computing. Moreover, the rise of remote work has expanded opportunities for data scientists globally, contributing to competitive salary packages, especially in tech, finance, and healthcare sectors.

About the dataset

In the rapidly evolving field of data science, understanding the trends and patterns in salaries is crucial for professionals and organizations alike. This dataset aims to shed light on the landscape of Data Science Salaries from 2020 to 2024. By analyzing salary data over this period, data enthusiasts, researchers, and industry professionals can gain valuable insights into salary trends, regional variations, and potential factors influencing compensation within the data science community. Dataset Structure

This dataset (data_science_salaries) covering from 2020 up to 2024 includes the following columns:

Column Name & Description

job_title --The job title or role associated with the reported salary. experience_level --The level of experience of the individual. employment_type --Indicates whether the employment is full-time, part-time, etc. work_models --Describes different working models (remote, on-site, hybrid). work_year-- The specific year in which the salary information was recorded. employee_residence --The residence location of the employee. salary --The reported salary in the original currency. salary_currency-- The currency in which the salary is denominated. salary_in_usd --The converted salary in US dollars. company_location --The geographic location of the employing organization. company_size --The size of the company, categorized by the number of employees.

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6599 entries, 0 to 6598
Data columns (total 11 columns):
 #   Column              Non-Null Count  Dtype 
---  ------              --------------  ----- 
 0   job_title           6599 non-null   object
 1   experience_level    6599 non-null   object
 2   employment_type     6599 non-null   object
 3   work_models         6599 non-null   object
 4   work_year           6599 non-null   int64 
 5   employee_residence  6599 non-null   object
 6   salary              6599 non-null   int64 
 7   salary_currency     6599 non-null   object
 8   salary_in_usd       6599 non-null   int64 
 9   company_location    6599 non-null   object
 10  company_size        6599 non-null   object
dtypes: int64(3), object(8)
memory usage: 567.2+ KB
```

```
df.shape
```

```
(6599, 11)
```

```
df.describe()
```

```
         work_year        salary  salary_in_usd
count  6599.000000  6.599000e+03    6599.000000
mean   2022.818457  1.792833e+05  145560.558569
std       0.674809  5.263722e+05   70946.838070
min    2020.000000  1.400000e+04   15000.000000
25%    2023.000000  9.600000e+04   95000.000000
50%    2023.000000  1.400000e+05  138666.000000
75%    2023.000000  1.875000e+05  185000.000000
max    2024.000000  3.040000e+07  750000.000000
```

```
df.describe(include=[object])
```

```
            job_title experience_level employment_type work_models  \
count            6599             6599            6599        6599
unique            132                4               4           3
top     Data Engineer     Senior-level       Full-time     On-site
freq             1307             4105            6552        3813

       employee_residence salary_currency company_location
company_size
count                6599            6599             6599
6599
unique                 87              22               75
3
top         United States             USD    United States
Medium
freq                 5305            5827             5354
5860
```

```
df.shape
```

```
(6599, 11)
```

```
df.columns
```

```
Index(['job_title', 'experience_level', 'employment_type',
'work_models',
       'work_year', 'employee_residence', 'salary', 'salary_currency',
       'salary_in_usd', 'company_location', 'company_size'],
      dtype='object')
```

```
df.dtypes
```

```
job_title             object
experience_level      object
employment_type       object
work_models           object
work_year              int64
employee_residence    object
salary                 int64
salary_currency       object
```

```
salary_in_usd          int64
company_location      object
company_size          object
dtype: object
```

```
df.isnull().sum()
```

```
job_title             0
experience_level      0
employment_type       0
work_models           0
work_year             0
employee_residence    0
salary                0
salary_currency       0
salary_in_usd         0
company_location      0
company_size          0
dtype: int64
```

```
df.mean
```

```
<bound method DataFrame.mean of                         job_title
experience_level employment_type work_models  \
0                 Data Engineer        Mid-level       Full-time
Remote
1                 Data Engineer        Mid-level       Full-time
Remote
2               Data Scientist     Senior-level       Full-time
Remote
3               Data Scientist     Senior-level       Full-time
Remote
4                 BI Developer        Mid-level       Full-time
On-site
...                         ...              ...             ...
...
6594        Staff Data Analyst      Entry-level        Contract
Hybrid
6595        Staff Data Analyst  Executive-level       Full-time
On-site
6596  Machine Learning Manager     Senior-level       Full-time
Hybrid
6597              Data Engineer        Mid-level       Full-time
Hybrid
6598            Data Scientist     Senior-level       Full-time
On-site

      work_year employee_residence  salary salary_currency
salary_in_usd  \
0          2024      United States  148100             USD
```

```
148100
1          2024      United States   98700             USD
98700
2          2024      United States  140032             USD
140032
3          2024      United States  100022             USD
100022
4          2024      United States  120000             USD
120000
...         ...                ...     ...             ...
...
6594       2020             Canada   60000             CAD
44753
6595       2020            Nigeria   15000             USD
15000
6596       2020             Canada  157000             CAD
117104
6597       2020            Austria   65000             EUR
74130
6598       2020            Austria   80000             EUR
91237

     company_location company_size
0       United States       Medium
1       United States       Medium
2       United States       Medium
3       United States       Medium
4       United States       Medium
...               ...          ...
6594           Canada        Large
6595           Canada       Medium
6596           Canada        Large
6597          Austria        Large
6598          Austria        Small

[6599 rows x 11 columns]>

df.sum

<bound method DataFrame.sum of                         job_title
experience_level employment_type work_models  \
0                 Data Engineer        Mid-level       Full-time
Remote
1                 Data Engineer        Mid-level       Full-time
Remote
2                Data Scientist     Senior-level       Full-time
Remote
3                Data Scientist     Senior-level       Full-time
Remote
4                  BI Developer        Mid-level       Full-time
```

```
On-site
...                          ...              ...             ...
...
6594        Staff Data Analyst      Entry-level        Contract
Hybrid
6595        Staff Data Analyst  Executive-level       Full-time
On-site
6596  Machine Learning Manager     Senior-level       Full-time
Hybrid
6597             Data Engineer        Mid-level       Full-time
Hybrid
6598            Data Scientist     Senior-level       Full-time
On-site

      work_year employee_residence  salary salary_currency
salary_in_usd  \
0          2024      United States  148100             USD
148100
1          2024      United States   98700             USD
98700
2          2024      United States  140032             USD
140032
3          2024      United States  100022             USD
100022
4          2024      United States  120000             USD
120000
...         ...                ...     ...             ...
...
6594       2020             Canada   60000             CAD
44753
6595       2020            Nigeria   15000             USD
15000
6596       2020             Canada  157000             CAD
117104
6597       2020            Austria   65000             EUR
74130
6598       2020            Austria   80000             EUR
91237

     company_location company_size
0       United States       Medium
1       United States       Medium
2       United States       Medium
3       United States       Medium
4       United States       Medium
...               ...          ...
6594           Canada        Large
6595           Canada       Medium
6596           Canada        Large
```

```
6597          Austria        Large
6598          Austria        Small

[6599 rows x 11 columns]>
```

```
df.duplicated().sum()
```

```
0
```

```
df.drop(columns=['salary','salary_currency'], axis =1 , inplace= True)
```

```
cat_data= df.select_dtypes(include =['object'])
cat_col=cat_data.columns
num_data= df.select_dtypes(include =['int64'])
num_col=num_data.columns
```

```
cat_data.describe().T
```

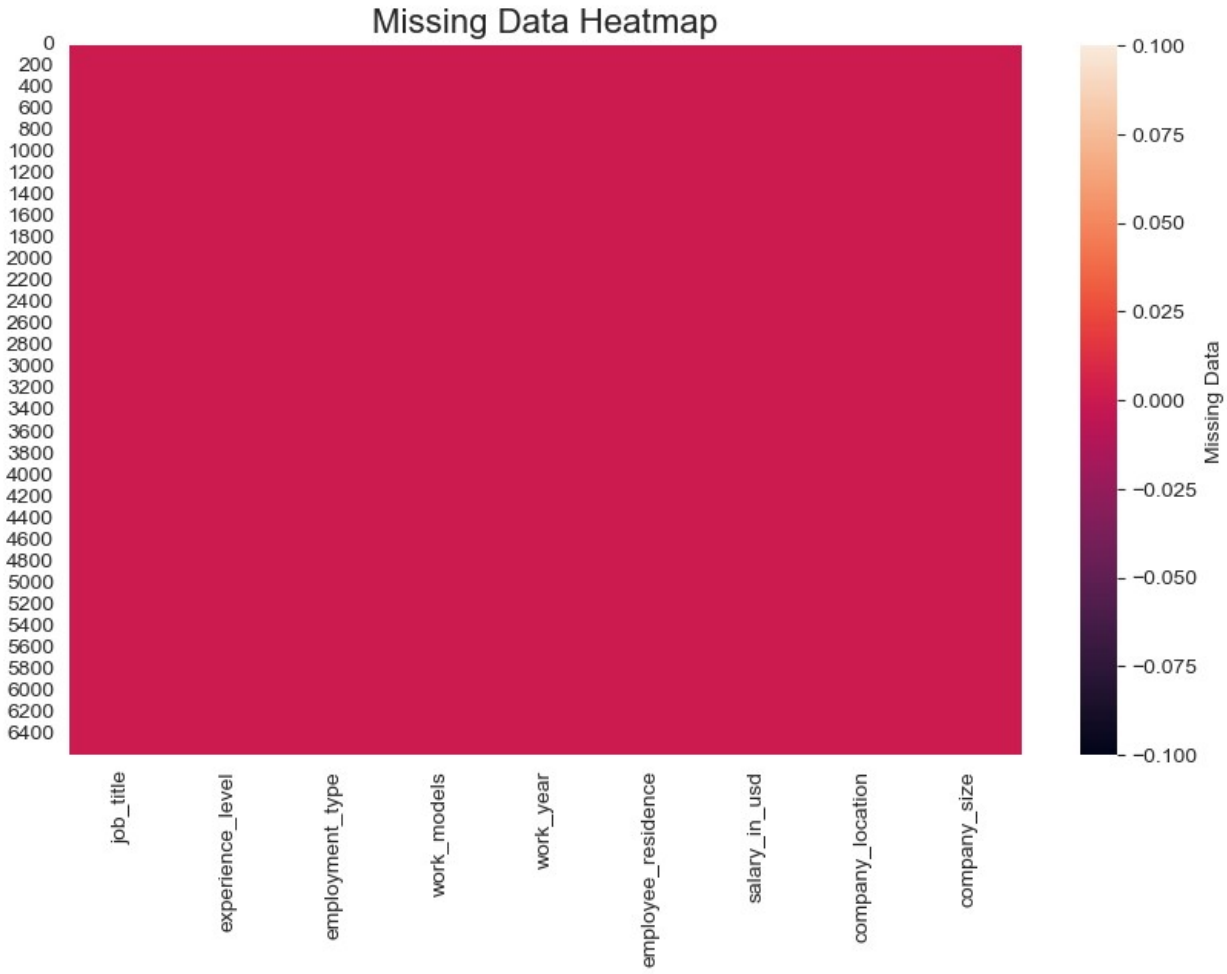
```
                    count unique            top  freq
job_title            6599    132  Data Engineer  1307
experience_level     6599      4   Senior-level  4105
employment_type      6599      4      Full-time  6552
work_models          6599      3        On-site  3813
employee_residence   6599     87  United States  5305
company_location     6599     75  United States  5354
company_size         6599      3         Medium  5860
```

```
num_data.describe()
```

```
         work_year  salary_in_usd
count  6599.000000    6599.000000
mean   2022.818457  145560.558569
std       0.674809   70946.838070
min    2020.000000   15000.000000
25%    2023.000000   95000.000000
50%    2023.000000  138666.000000
75%    2023.000000  185000.000000
max    2024.000000  750000.000000
```

```
def missing(df):
    miss = pd.DataFrame(columns=['Column', 'Value', 'Percentage'])  # 
Initialize the DataFrame
    for col in df.columns:
        value = df[col].isnull().sum()
        percentage = (value / len(df)) * 100
        new_row = pd.DataFrame({'Column': [col], 'Value': [value], 
'Percentage': [percentage]})
        miss = pd.concat([miss, new_row], ignore_index=True)  # Use 
pd.concat instead of .append
    return miss
```

```
plt.figure(figsize=(10,6))
sns.heatmap(df.isnull(),cbar=True,    cbar_kws={'label': 'Missing
Data'},  cmap= 'rocket')
plt.title("Missing Data Heatmap", fontsize=16)
plt.show()
```



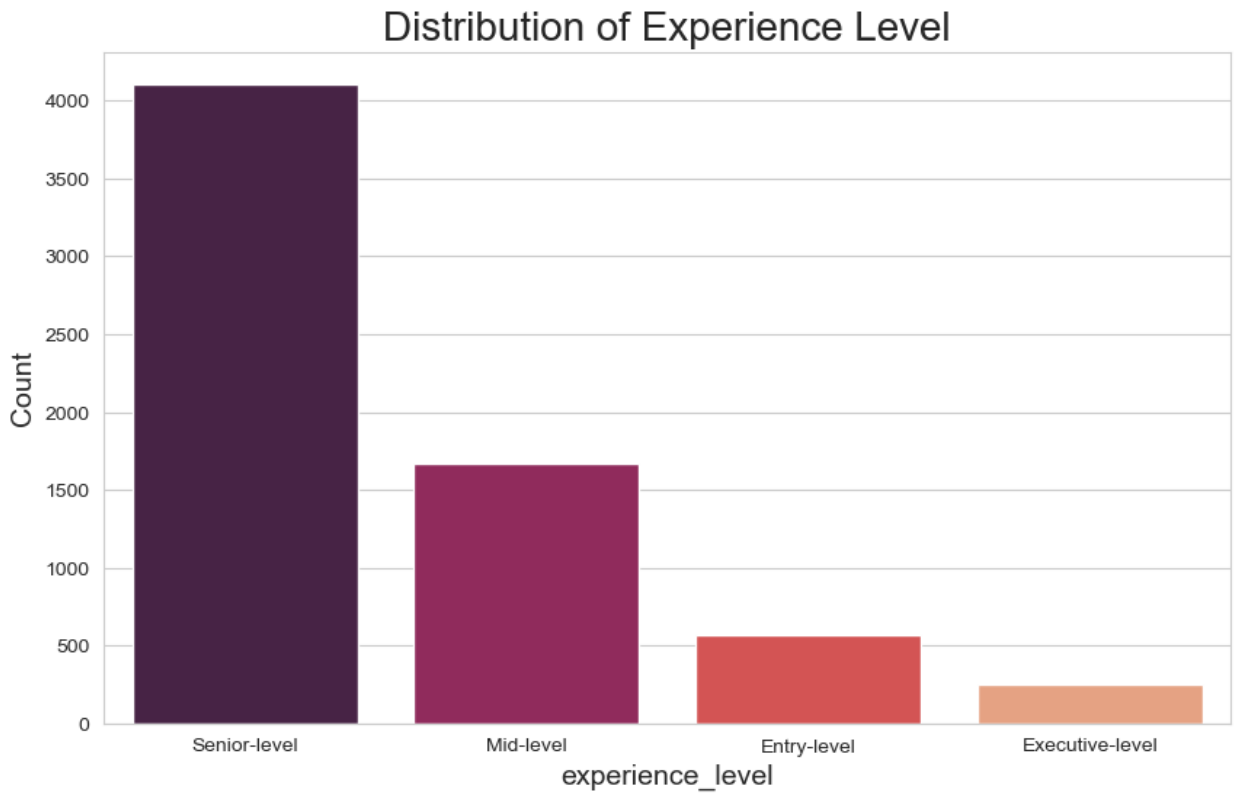

```
df.duplicated().sum()

0

df['experience_level'] = df['experience_level'].replace('EN','Junior')
df['experience_level'] =
df['experience_level'].replace('MI','Intermediate')
df['experience_level'] = df['experience_level'].replace('SE','Expert')
df['experience_level'] =
df['experience_level'].replace('EX','Director')

df['employment_type'] =
df['employment_type'].replace('PT','Part_time')
df['employment_type'] =
```

```
df['employment_type'].replace('FT','Full_time')
df['employment_type'] = df['employment_type'].replace('CT','Contract')
df['employment_type'] =
df['employment_type'].replace('FL','Freelance')

df['company_size'] = df['company_size'].replace('S','Small')
df['company_size'] = df['company_size'].replace('M','Medium')
df['company_size'] = df['company_size'].replace('L','Large')

df
```

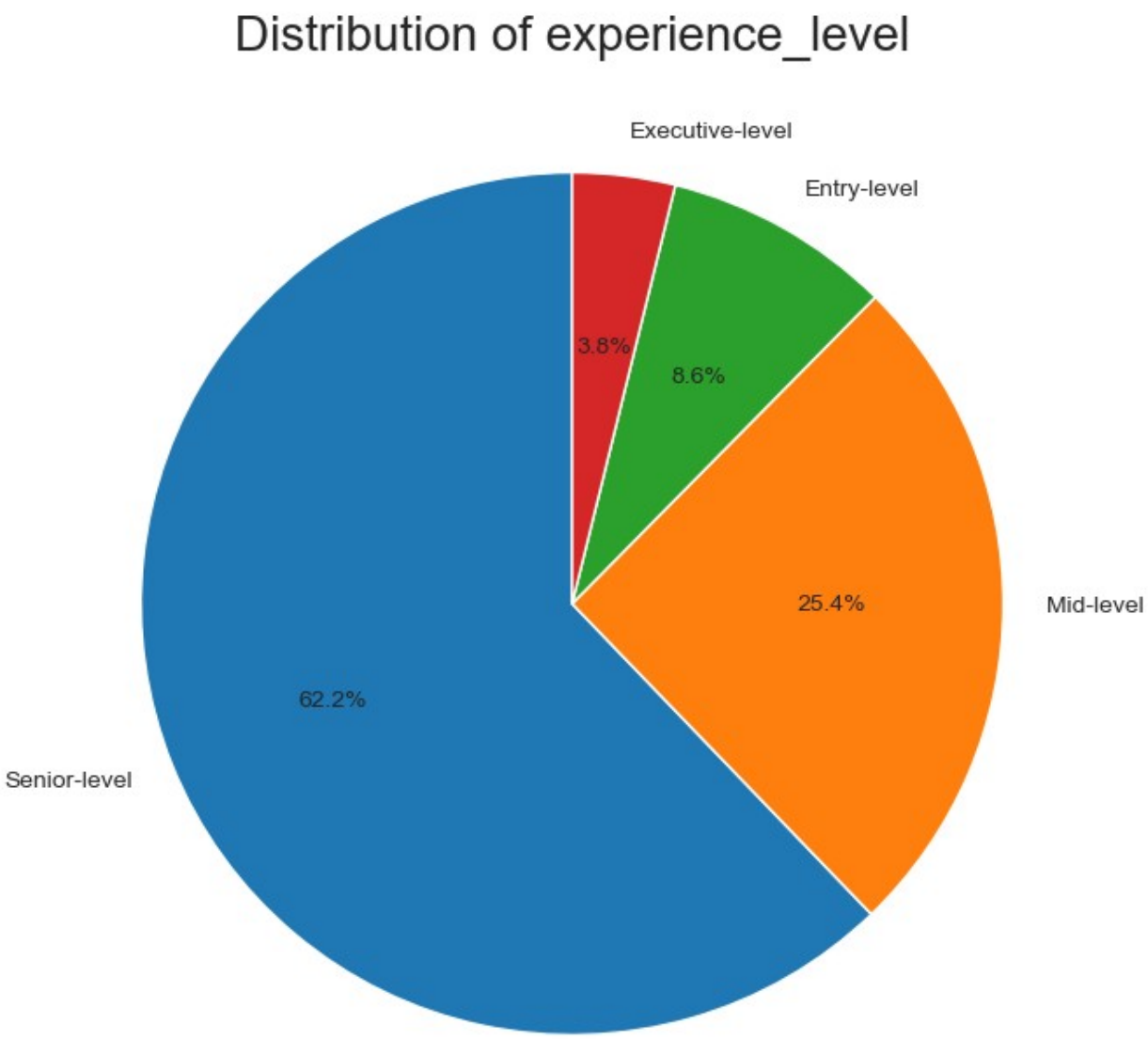
```
                    job_title experience_level employment_type
work_models  \
0                Data Engineer        Mid-level       Full-time
Remote
1                Data Engineer        Mid-level       Full-time
Remote
2               Data Scientist     Senior-level       Full-time
Remote
3               Data Scientist     Senior-level       Full-time
Remote
4                 BI Developer        Mid-level       Full-time
On-site
...                        ...              ...             ...
...
6594        Staff Data Analyst      Entry-level        Contract
Hybrid
6595        Staff Data Analyst  Executive-level       Full-time
On-site
6596  Machine Learning Manager     Senior-level       Full-time
Hybrid
6597             Data Engineer        Mid-level       Full-time
Hybrid
6598            Data Scientist     Senior-level       Full-time
On-site

      work_year employee_residence  salary_in_usd company_location  \
0          2024      United States         148100    United States
1          2024      United States          98700    United States
2          2024      United States         140032    United States
3          2024      United States         100022    United States
4          2024      United States         120000    United States
...         ...                ...            ...              ...
6594       2020             Canada          44753           Canada
6595       2020            Nigeria          15000           Canada
6596       2020             Canada         117104           Canada
6597       2020            Austria          74130          Austria
6598       2020            Austria          91237          Austria

     company_size
```

```
0          Medium
1          Medium
2          Medium
3          Medium
4          Medium
...           ...
6594        Large
6595       Medium
6596        Large
6597        Large
6598        Small

[6599 rows x 9 columns]
```

```
counts = df['experience_level'].value_counts()
top_categories = counts.head(10).index
plt.figure(figsize=(10, 6))
sns.countplot(data=df, x='experience_level', palette='rocket',
order=top_categories)
plt.title(f"Distribution of Experience Level", fontsize=20)
# plt.xticks(rotation=45)
plt.xlabel('experience_level', fontsize=14)
plt.ylabel("Count", fontsize=14)
plt.show()
```

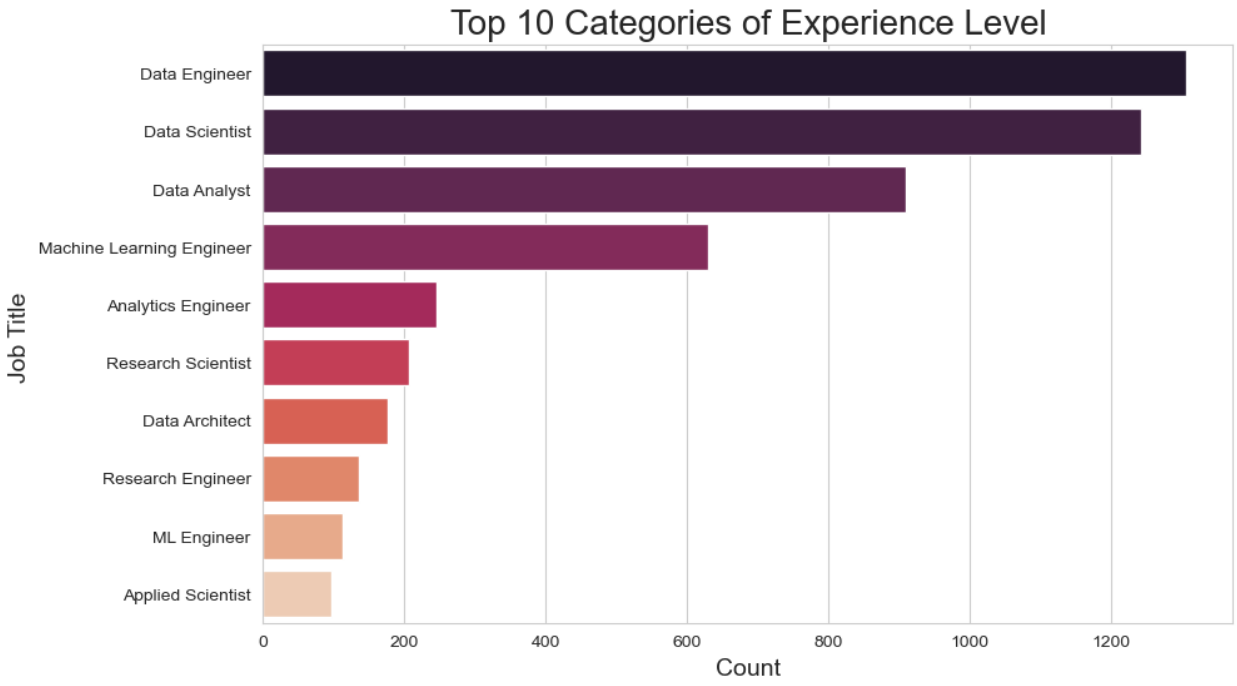


```
counts = df['experience_level'].value_counts()
top_categories = counts.head(10)
plt.figure(figsize=(8,8))
plt.pie(top_categories , labels = top_categories.index ,
        autopct = '%1.1f%%',startangle=90,colors=plt.cm.tab10.colors)
plt.title(f"Distribution of experience_level", fontsize=20)
plt.xticks(rotation=45)
plt.show()
```

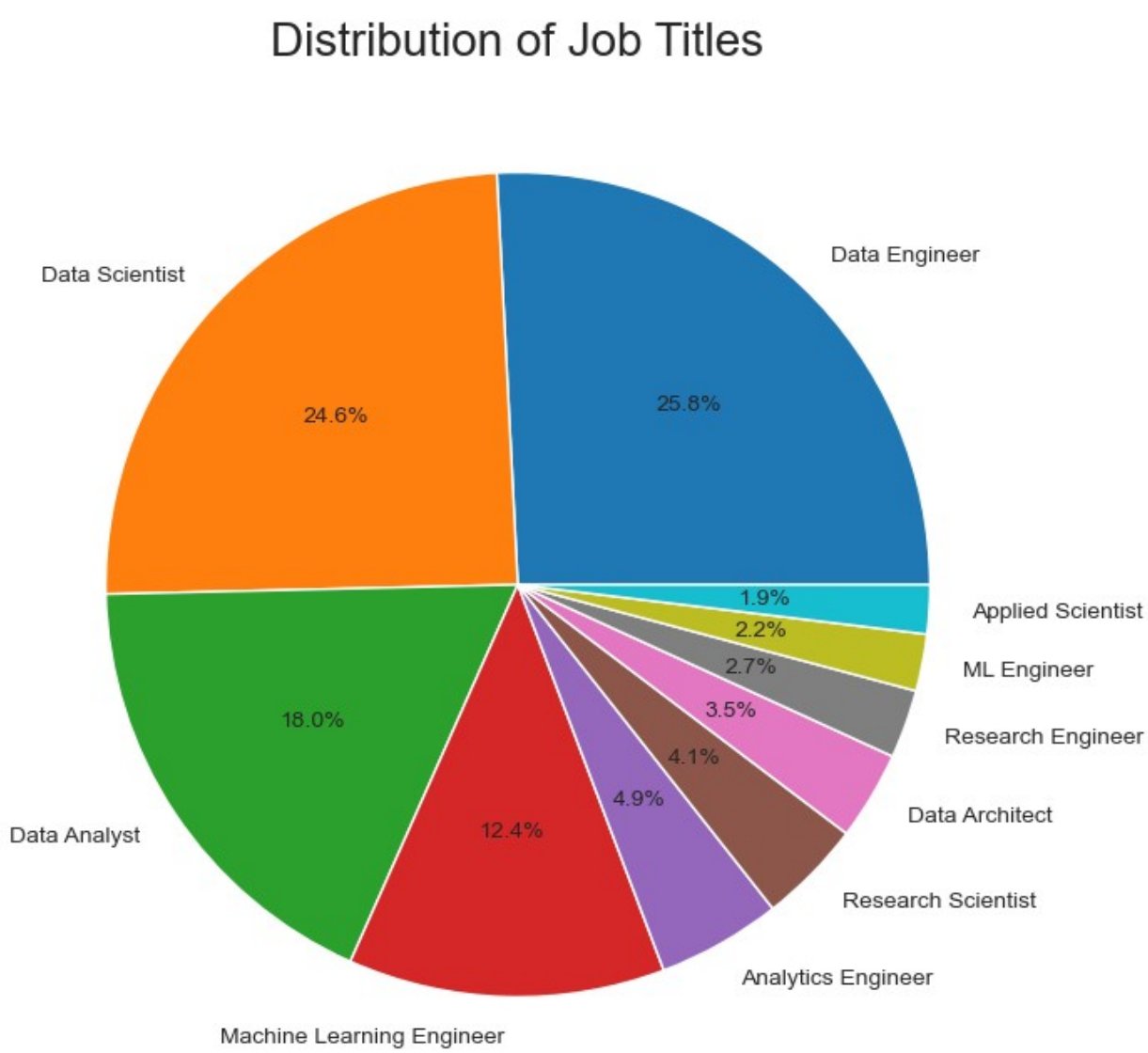



```
counts = df['job_title'].value_counts()
top_categories = counts.head(10).index
plt.figure(figsize=(10, 6))
```

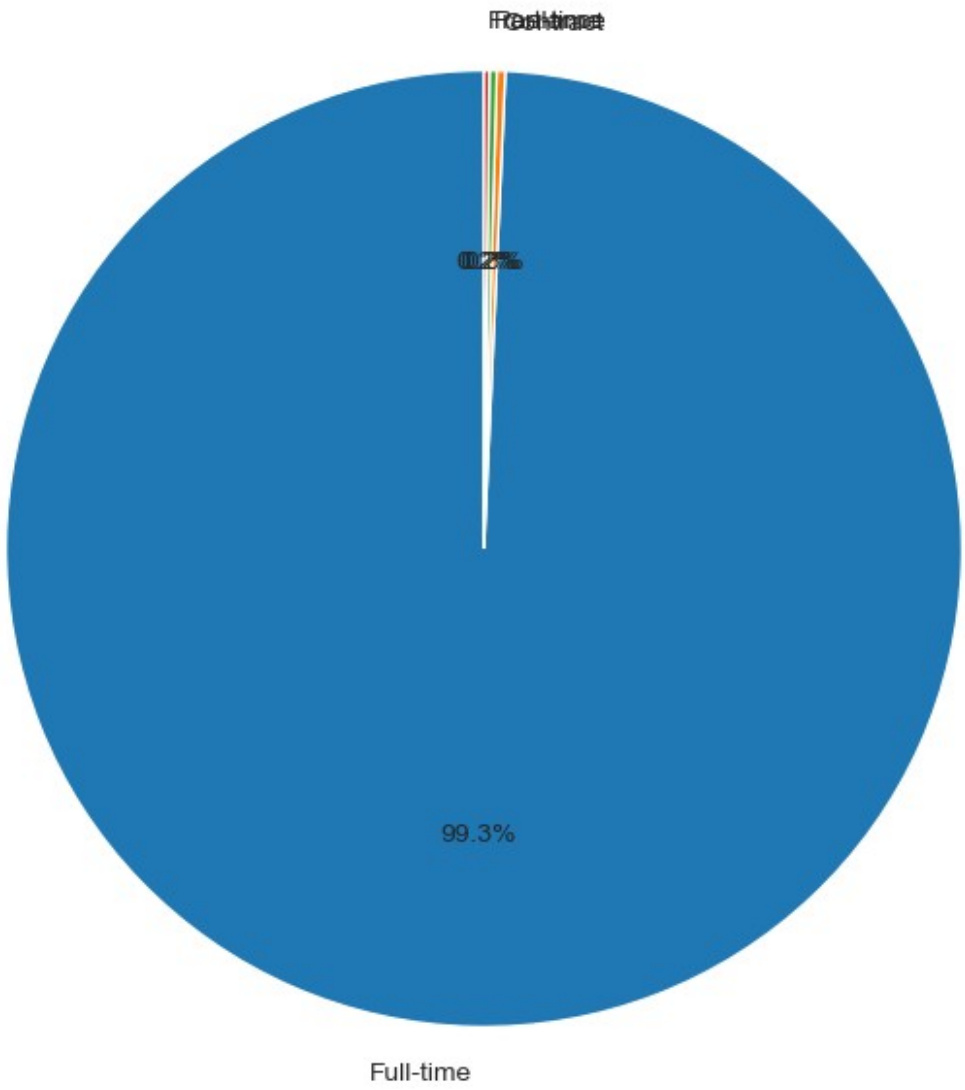
```
sns.countplot(data=df, y='job_title', palette='rocket',orient ='v',
order=top_categories)
plt.title(f"Top 10 Categories of Experience Level", fontsize=20)
plt.ylabel('Job Title ', fontsize=14)
plt.xlabel("Count", fontsize=14)
plt.show()
```



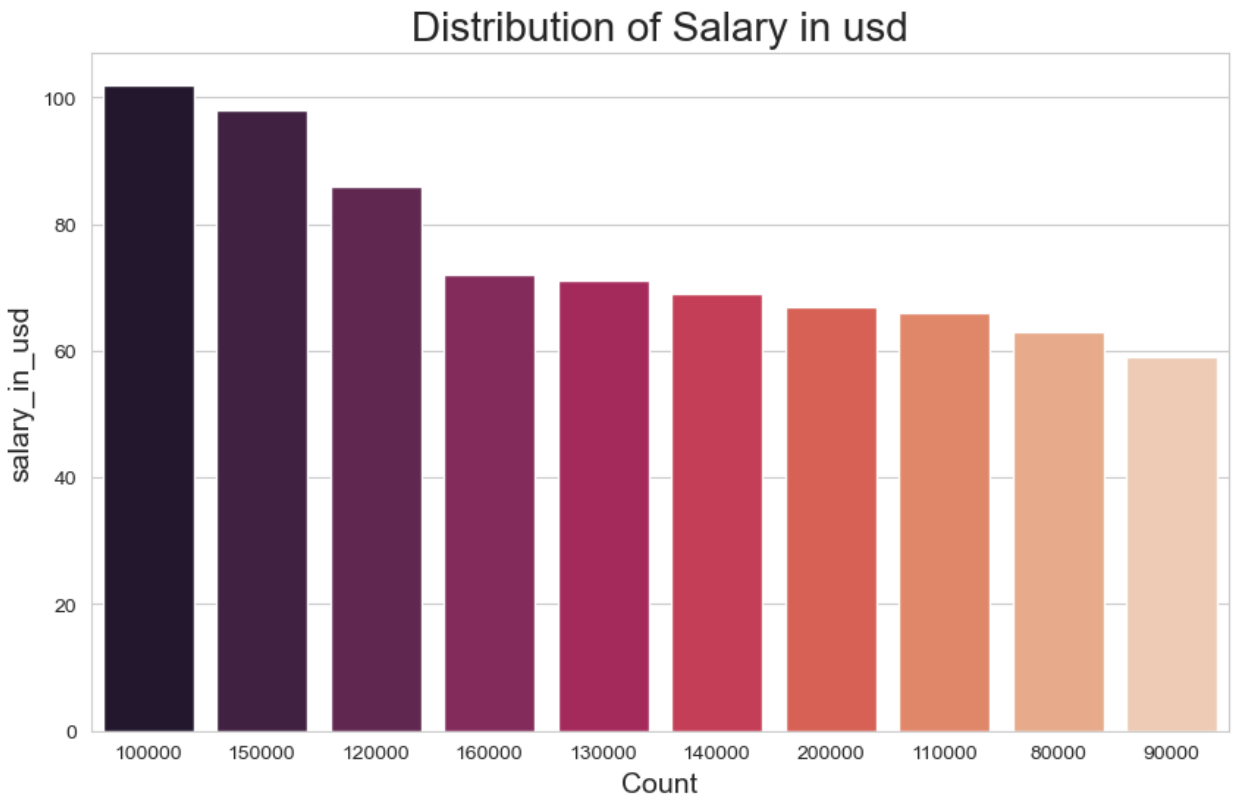

```
counts = df['job_title'].value_counts()
top_categories = counts.head(10)
plt.figure(figsize=(8,8))
plt.pie(top_categories , labels = top_categories.index ,
        autopct = '%1.1f%%')
plt.title(f"Distribution of Job Titles", fontsize=20)
plt.xticks(rotation=45)
plt.show()
```



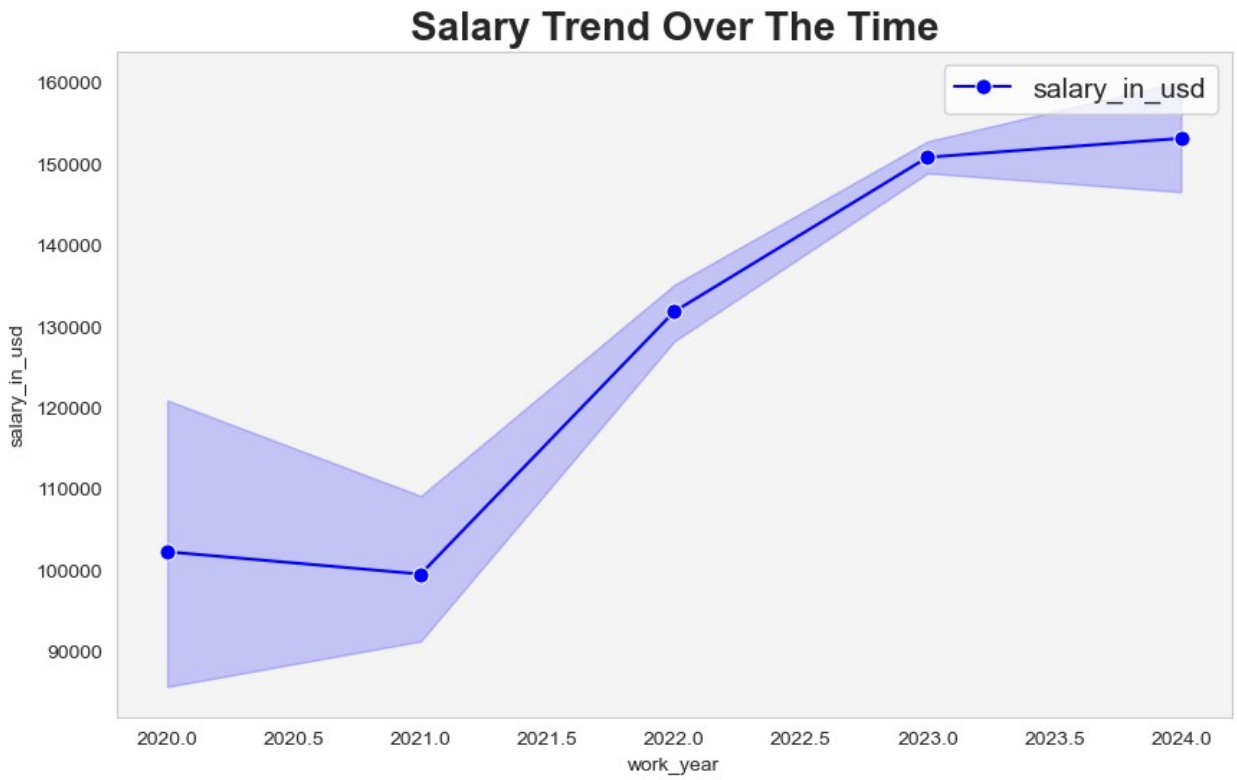

```
counts = df['employment_type'].value_counts()
top_categories = counts.head(10)
plt.figure(figsize=(8,8))
plt.pie(top_categories , labels = top_categories.index ,
        autopct = '%1.1f%%',startangle=90,colors=plt.cm.tab10.colors)
plt.title(f"Distribution of Employment type", fontsize=20)
plt.xticks(rotation=45)
plt.show()
```



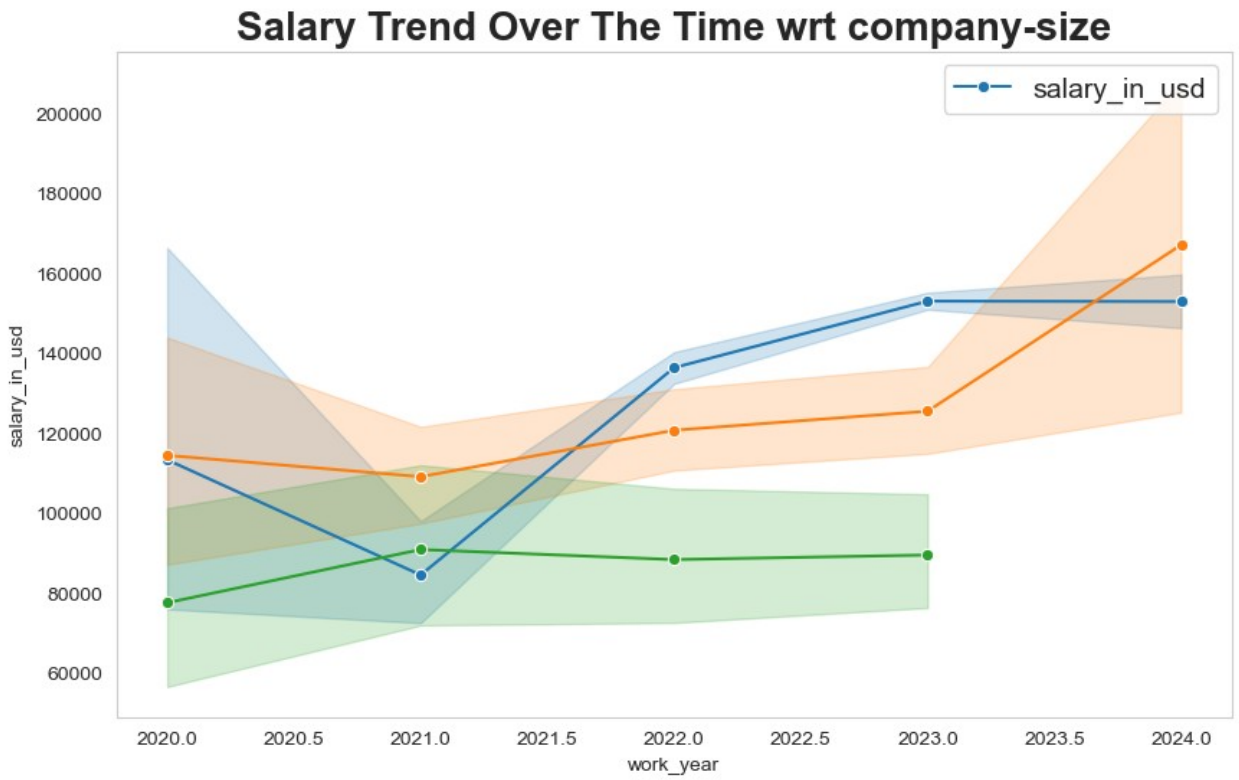

```
counts = df['salary_in_usd'].value_counts()
top_categories = counts.head(10).index
plt.figure(figsize=(10, 6))
sns.countplot(data=df, x='salary_in_usd', palette='rocket',orient
='v', order=top_categories)
plt.title(f"Distribution of Salary in usd", fontsize=20)
plt.ylabel('salary_in_usd', fontsize=14)
plt.xlabel("Count", fontsize=14)
plt.show()
```



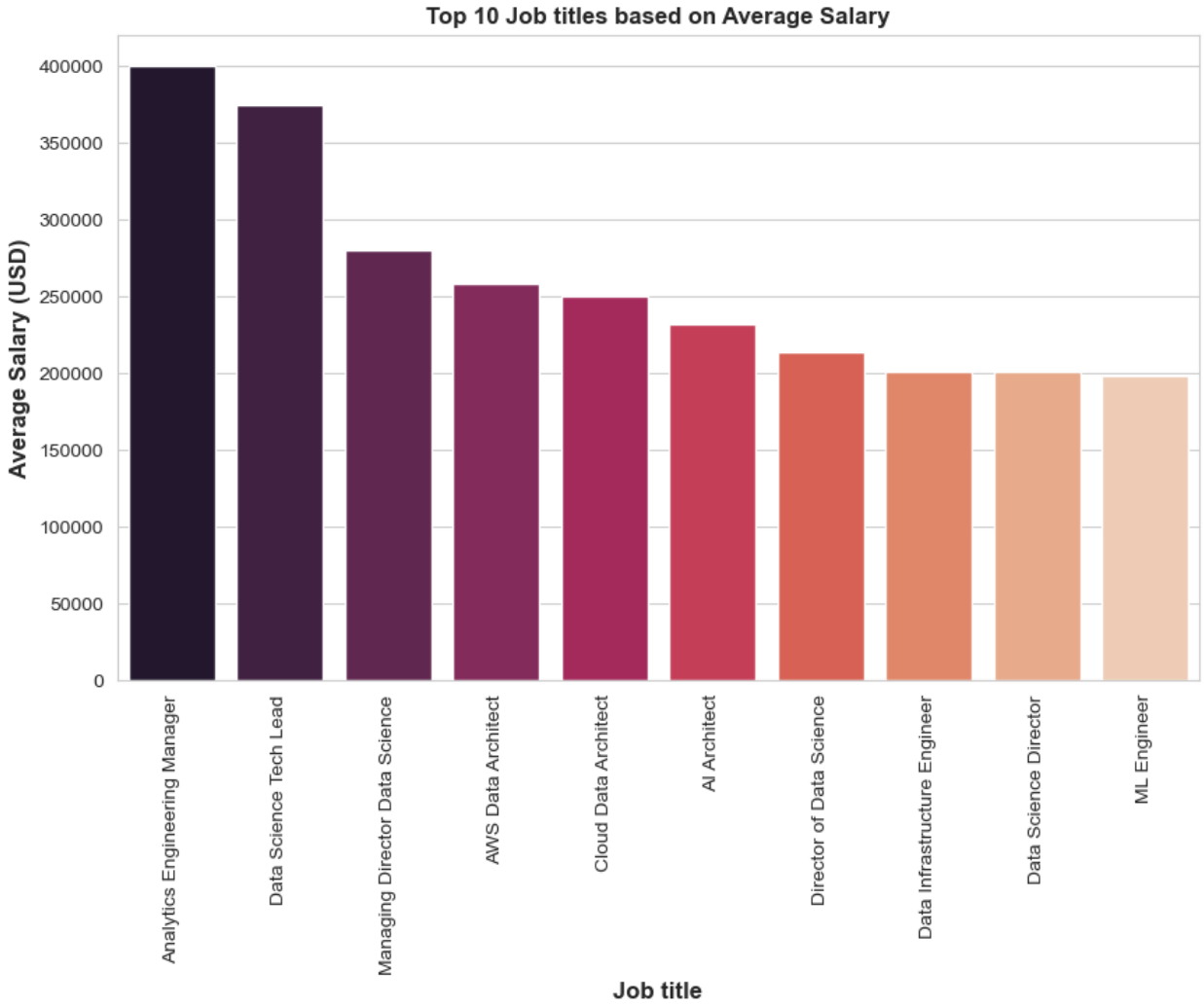

```
plt.figure(figsize = (10,6))
salary_trend = df[['salary_in_usd', 'work_year']].sort_values(by =
'work_year')
p = sns.lineplot(data =salary_trend ,x = 'work_year', y =
'salary_in_usd', marker = 'o', color='Blue', markersize=8 )
plt.title('Salary Trend Over The Time', fontsize=20,
fontweight='bold')
p.set_facecolor("#f4f4f4")
plt.legend(['salary_in_usd'], loc='best', fontsize=14)
p.grid(False)
plt.show()
```



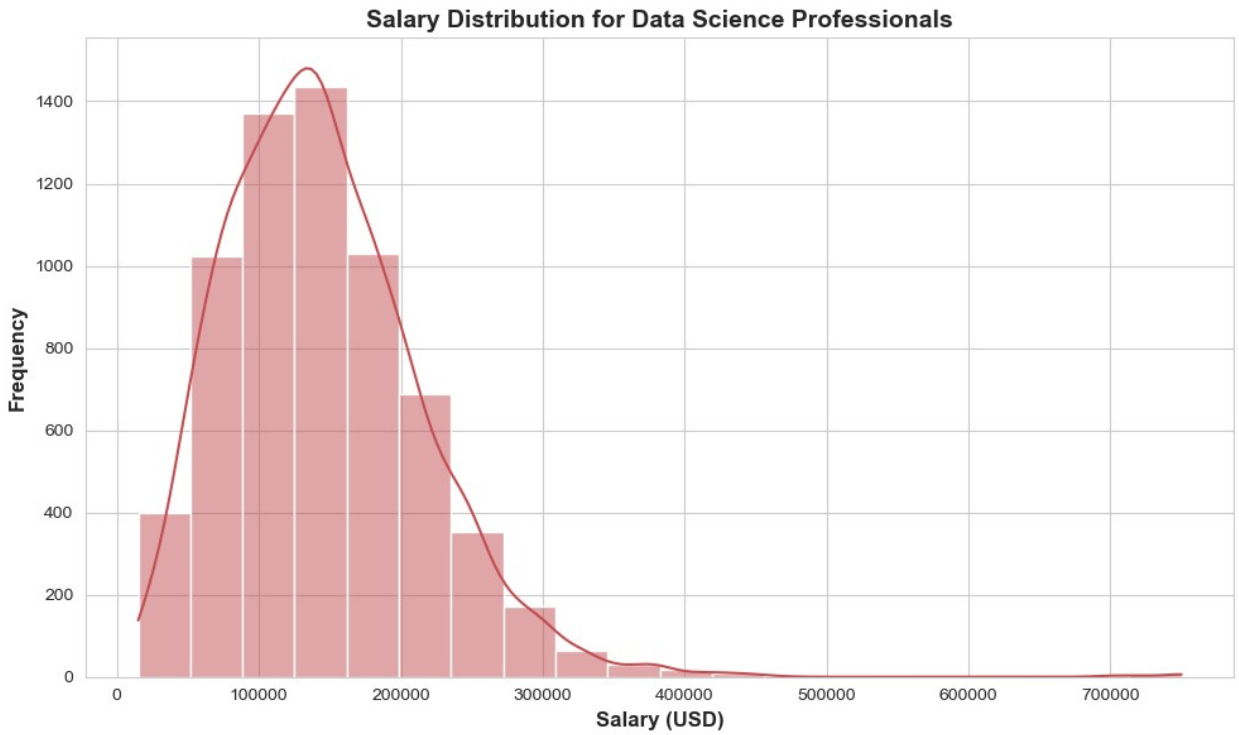

```
plt.figure(figsize = (10,6))
p = sns.lineplot(data =df ,x = 'work_year', y = 'salary_in_usd', hue =
'company_size',marker = 'o', color='purple')
plt.title('Salary Trend Over The Time wrt company-size', fontsize=20,
fontweight='bold')
plt.legend(['salary_in_usd'], loc='best', fontsize=14)
p.grid(False)
plt.show()
```




```
exp_salary = df.groupby('job_title')
['salary_in_usd'].mean().sort_values(ascending = False).head(10)
plt.figure(figsize = (10,6))
ax = sns.barplot(x = exp_salary.index, y = exp_salary.values, palette
= 'rocket')
plt.title('Top 10 Job titles based on Average Salary', fontsize=12,
fontweight='bold')
plt.xticks(rotation = 90)
plt.xlabel('Job title', fontsize=12, fontweight='bold')
plt.ylabel('Average Salary (USD)', fontsize=12, fontweight='bold')
plt.show()
```




```
sns.set_style("whitegrid")
fig, ax = plt.subplots(figsize=(10, 6))
sns.histplot(df['salary_in_usd'], kde=True, color='#c44e52', bins=20,
ax=ax)
ax.set_title('Salary Distribution for Data Science Professionals',
fontsize=14, fontweight='bold')
ax.set_xlabel('Salary (USD)', fontsize=12, fontweight='bold')
ax.set_ylabel('Frequency', fontsize=12, fontweight='bold')
plt.tight_layout()
plt.show()
```



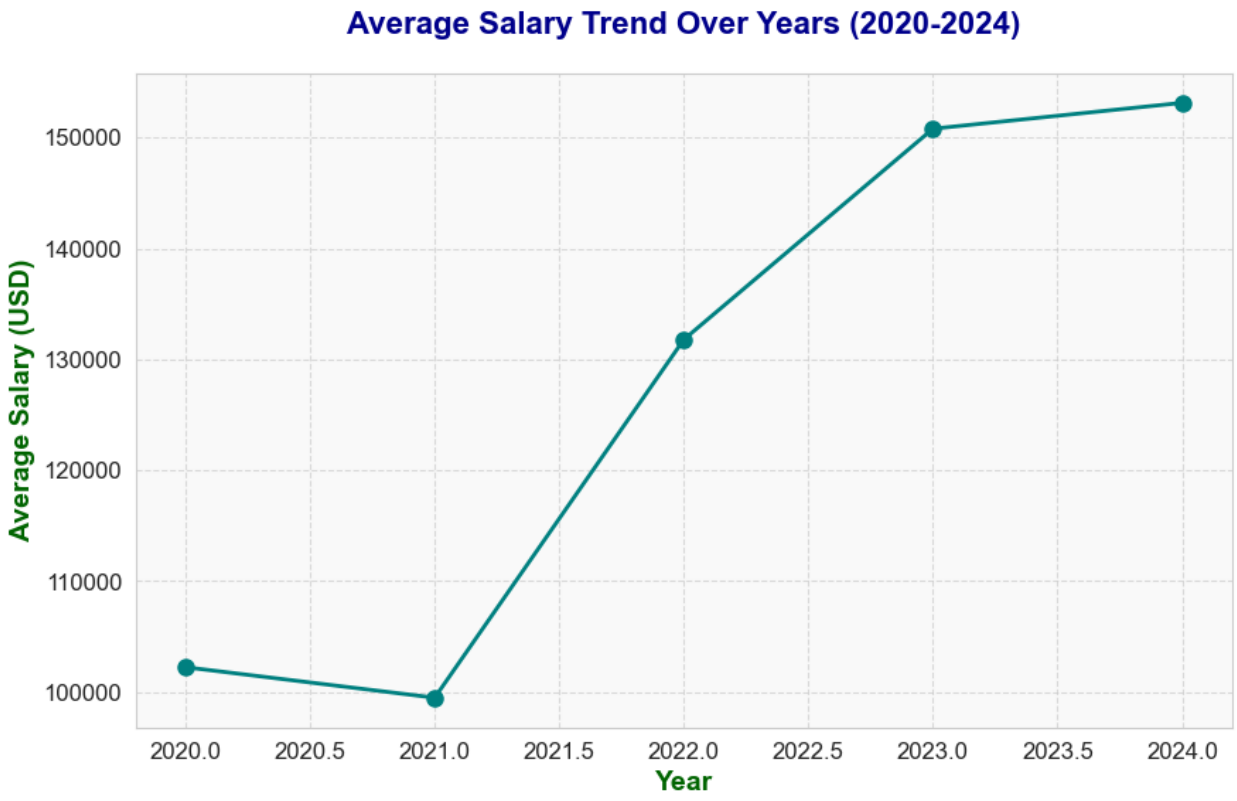

```
# Categorical columns distribution
categorical_cols = ['job_title', 'experience_level',
'employment_type', 'work_models', 'employee_residence',
'company_location', 'company_size']
for col in categorical_cols:
    print(df[col].value_counts())
```

```
job_title
Data Engineer                1307
Data Scientist               1243
Data Analyst                  910
Machine Learning Engineer     629
Analytics Engineer            246
                             ...
Deep Learning Researcher        1
Power BI Developer              1
Marketing Data Scientist        1
AI Product Manager              1
Sales Data Analyst              1
Name: count, Length: 132, dtype: int64
experience_level
Senior-level       4105
Mid-level          1675
Entry-level         565
```

```
Executive-level     254
Name: count, dtype: int64
employment_type
Full-time    6552
Contract       19
Part-time      16
Freelance      12
Name: count, dtype: int64
work_models
On-site    3813
Remote     2561
Hybrid      225
Name: count, dtype: int64
employee_residence
United States     5305
United Kingdom     401
Canada             241
Germany             71
India               70
                   ...
Georgia               1
Israel                1
Qatar                 1
Peru                  1
Honduras              1
Name: count, Length: 87, dtype: int64
company_location
United States             5354
United Kingdom             408
Canada                     243
Germany                     78
Spain                       63
                           ...
Armenia                      1
Bosnia and Herzegovina       1
Qatar                        1
Ecuador                      1
Honduras                     1
Name: count, Length: 75, dtype: int64
company_size
Medium    5860
Large      569
Small      170
Name: count, dtype: int64

# Disable scientific notation in pandas
pd.set_option('display.float_format', '{:.2f}'.format)

# Numerical columns summary without scientific notation
print(df[['salary', 'salary_in_usd']].describe())
```

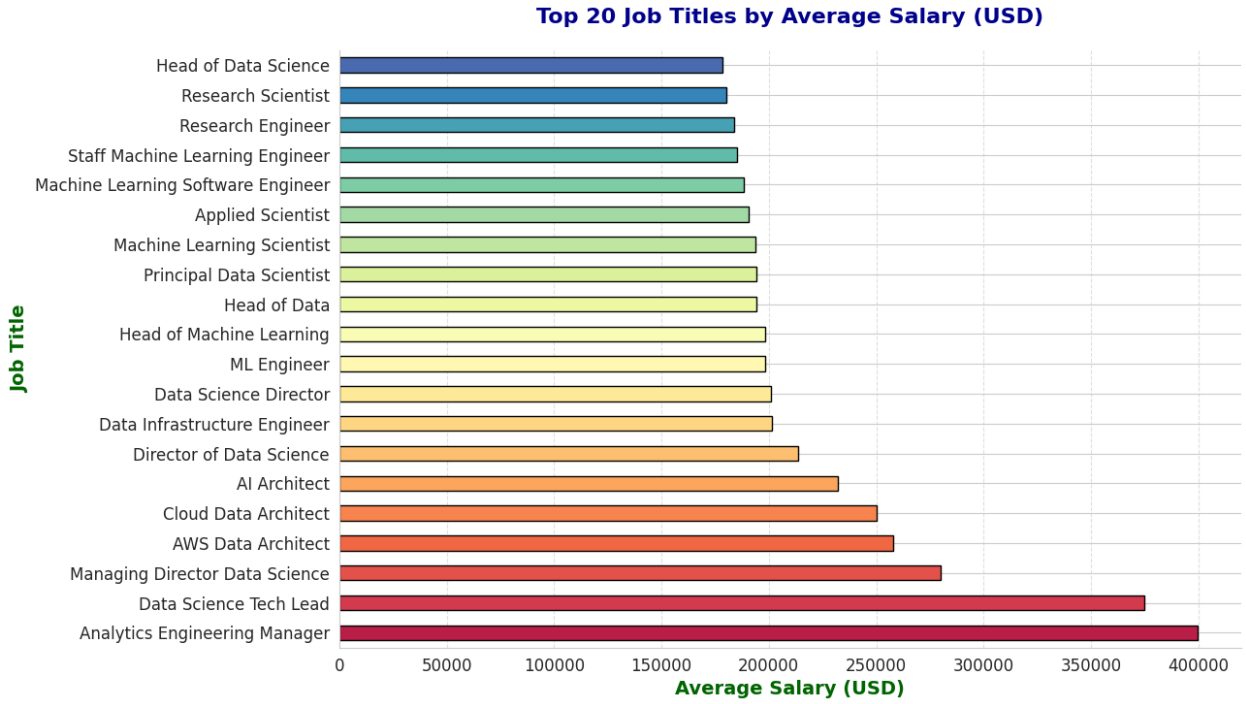
```
             salary  salary_in_usd
count     6599.00        6599.00
mean    179283.26      145560.56
std     526372.24       70946.84
min      14000.00       15000.00
25%      96000.00       95000.00
50%     140000.00      138666.00
75%     187500.00      185000.00
max   30400000.00      750000.00
```

```
sns.set_style("whitegrid")

plt.figure(figsize=(10, 6))

# Plot the salary trend with line and marker styles
df.groupby('work_year')['salary_in_usd'].mean().plot(kind='line',
marker='o', markersize=8, linestyle='-', linewidth=2, color='teal')

# Add title and labels
plt.title('Average Salary Trend Over Years (2020-2024)', fontsize=16,
fontweight='bold', color='darkblue', pad=20)
plt.ylabel('Average Salary (USD)', fontsize=14, fontweight='bold',
color='darkgreen')
plt.xlabel('Year', fontsize=14, fontweight='bold', color='darkgreen')

# Customize ticks
plt.xticks(fontsize=12)
plt.yticks(fontsize=12)

# Add gridlines
plt.grid(True, which='both', linestyle='--', alpha=0.7)

# Add background color
plt.gca().set_facecolor('#f9f9f9')

# Show the plot
plt.show()
```



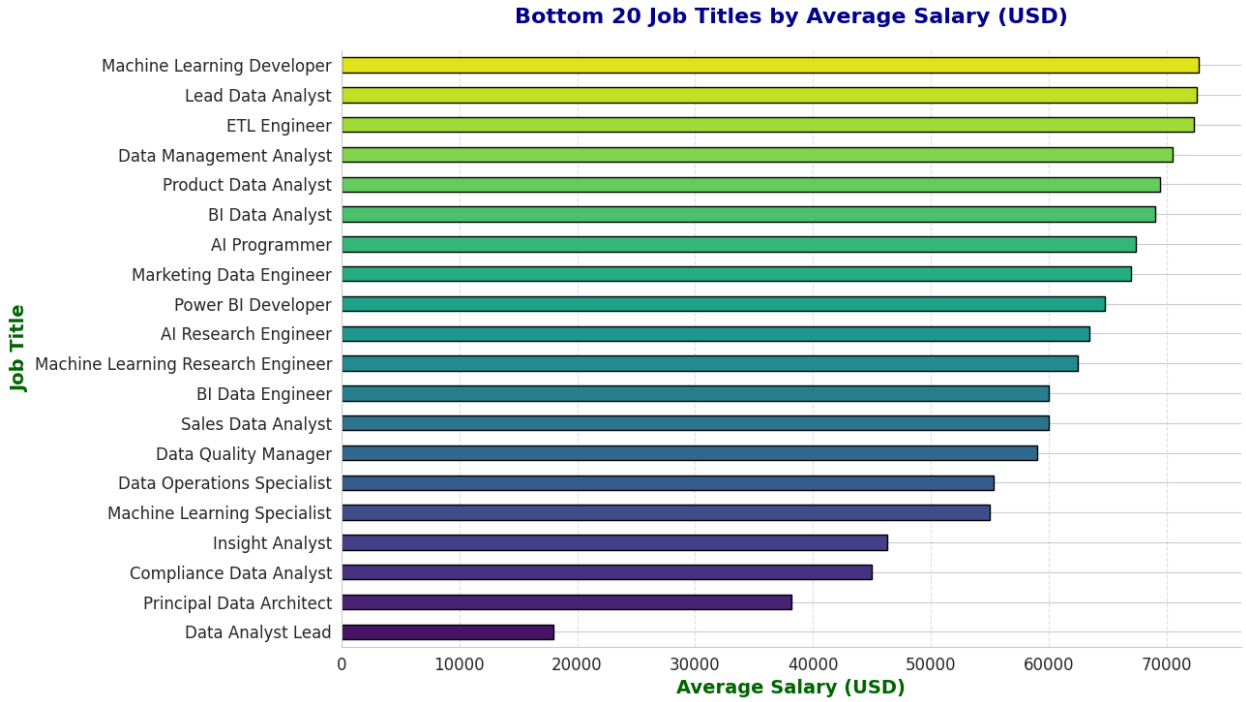

```
# Calculate the top 20 job titles by average salary
top_20_jobs = df.groupby('job_title')
['salary_in_usd'].mean().sort_values(ascending=False).head(20)


# color palette
colors = sns.color_palette("Spectral", len(top_20_jobs))

# Create the plot
plt.figure(figsize=(12, 8))
top_20_jobs.plot(kind='barh', color=colors, edgecolor='black')

# Title and labels
plt.title('Top 20 Job Titles by Average Salary (USD)', fontsize=16,
fontweight='bold', color='darkblue', pad=20)
plt.xlabel('Average Salary (USD)', fontsize=14, fontweight='bold',
color='darkgreen')
plt.ylabel('Job Title', fontsize=14, fontweight='bold',
color='darkgreen')

# Adding gridlines for better readability
plt.grid(axis='x', linestyle='--', alpha=0.6)

# Rotate x-axis labels for better readability and layout
plt.xticks(fontsize=12)
```

```
plt.yticks(fontsize=12)

# Remove top and right spines
sns.despine()

# Show the plot
plt.show()
```



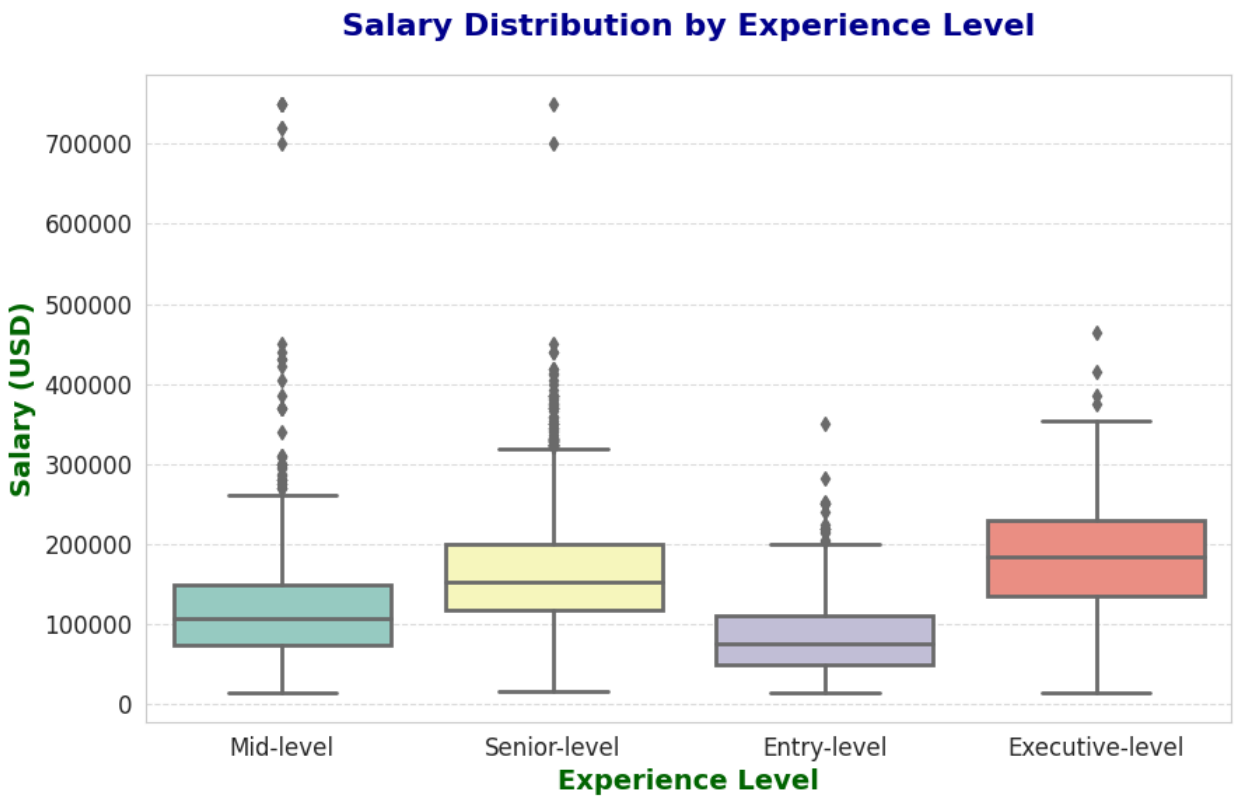

```
# Calculate the bottom 20 job titles by average salary
bottom_20_jobs = df.groupby('job_title')
['salary_in_usd'].mean().sort_values(ascending=True).head(20)

colors = sns.color_palette("viridis", len(bottom_20_jobs))

# Create the plot
plt.figure(figsize=(12, 8))
bottom_20_jobs.plot(kind='barh', color=colors, edgecolor='black')

# Correct title and labels
plt.title('Bottom 20 Job Titles by Average Salary (USD)', fontsize=16,
fontweight='bold', color='darkblue', pad=20)
plt.xlabel('Average Salary (USD)', fontsize=14, fontweight='bold',
color='darkgreen')
plt.ylabel('Job Title', fontsize=14, fontweight='bold',
color='darkgreen')

# Adding gridlines
plt.grid(axis='x', linestyle='--', alpha=0.6)
```

```
# Adjust axis tick labels
plt.xticks(fontsize=12)
plt.yticks(fontsize=12)

# Remove top and right spines
sns.despine()

# Show the plot
plt.show()
```

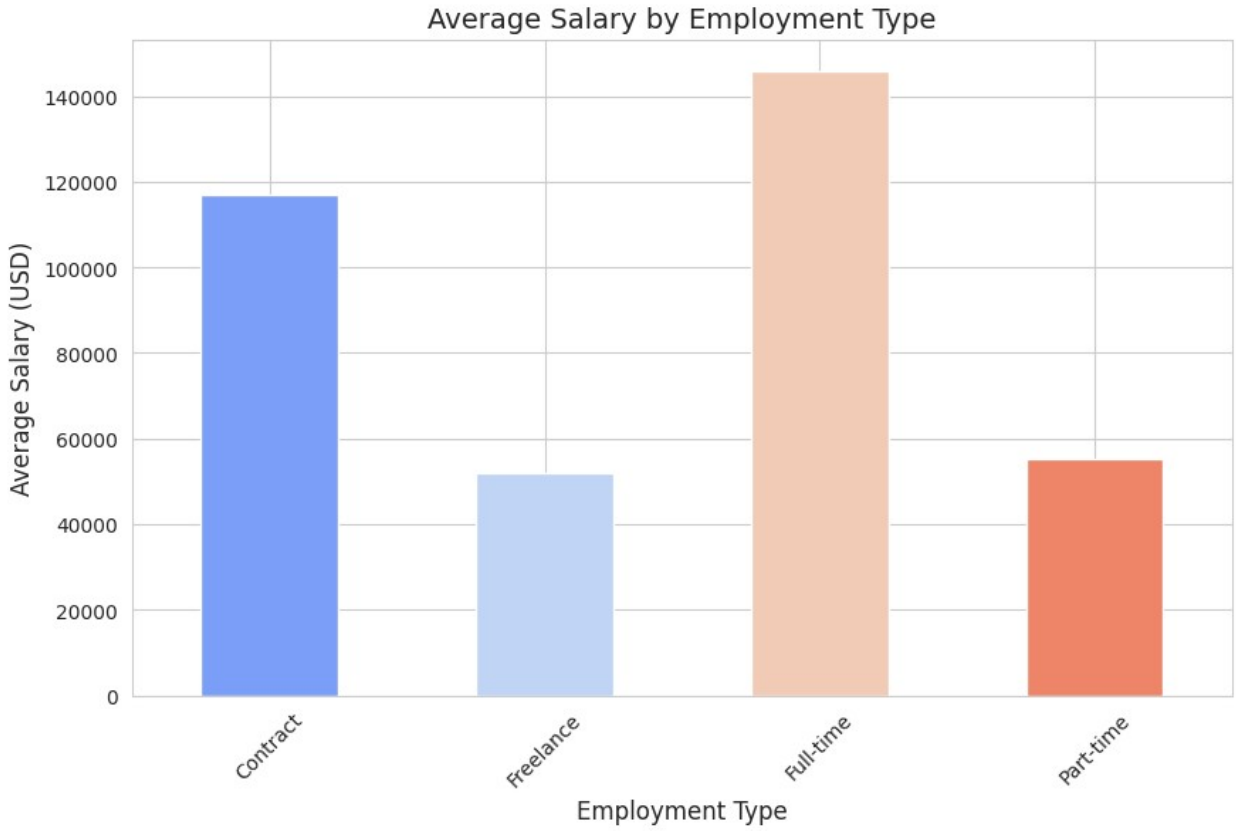



```
plt.figure(figsize=(10, 6))
sns.boxplot(x='experience_level', y='salary_in_usd', data=df,
palette="Set3", linewidth=2, fliersize=5)

# title and labels
plt.title('Salary Distribution by Experience Level', fontsize=16,
fontweight='bold', color='darkblue', pad=20)
plt.xlabel('Experience Level', fontsize=14, fontweight='bold',
color='darkgreen')
plt.ylabel('Salary (USD)', fontsize=14, fontweight='bold',
color='darkgreen')

# Adjust tick parameters
plt.xticks(fontsize=12)
plt.yticks(fontsize=12)

# Add gridlines for the y-axis
```

```
plt.grid(axis='y', linestyle='--', alpha=0.7)

# Show the plot
plt.show()
```



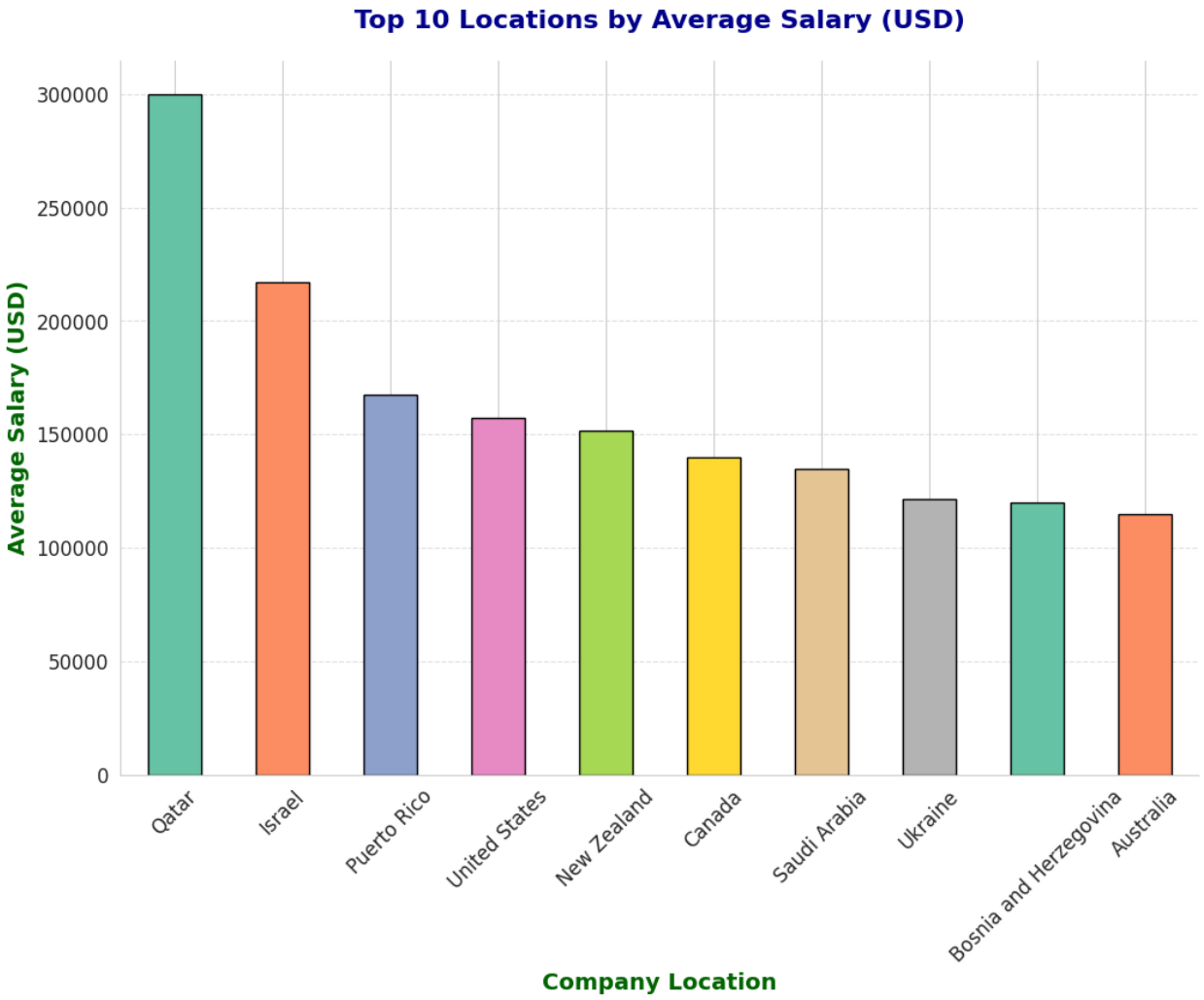

```
# Use a seaborn color palette
colors = sns.color_palette("coolwarm",
len(df['employment_type'].unique()))

# Employment type salary comparison
df.groupby('employment_type')['salary_in_usd'].mean().plot(kind='bar',
color=colors, figsize=(10, 6))

plt.title('Average Salary by Employment Type', fontsize=14)
plt.ylabel('Average Salary (USD)', fontsize=12)
plt.xlabel('Employment Type', fontsize=12)
plt.xticks(rotation=45)
plt.show()
```




```
colors = sns.color_palette("husl", len(df['work_models'].unique()))

# Work models salary comparison
df.groupby('work_models')['salary_in_usd'].mean().plot(kind='bar',
color=colors, figsize=(10, 6))

plt.title('Average Salary by Work Models', fontsize=14)
plt.ylabel('Average Salary (USD)', fontsize=12)
plt.xlabel('Work Models', fontsize=12)
plt.xticks(rotation=45)
plt.show()
```

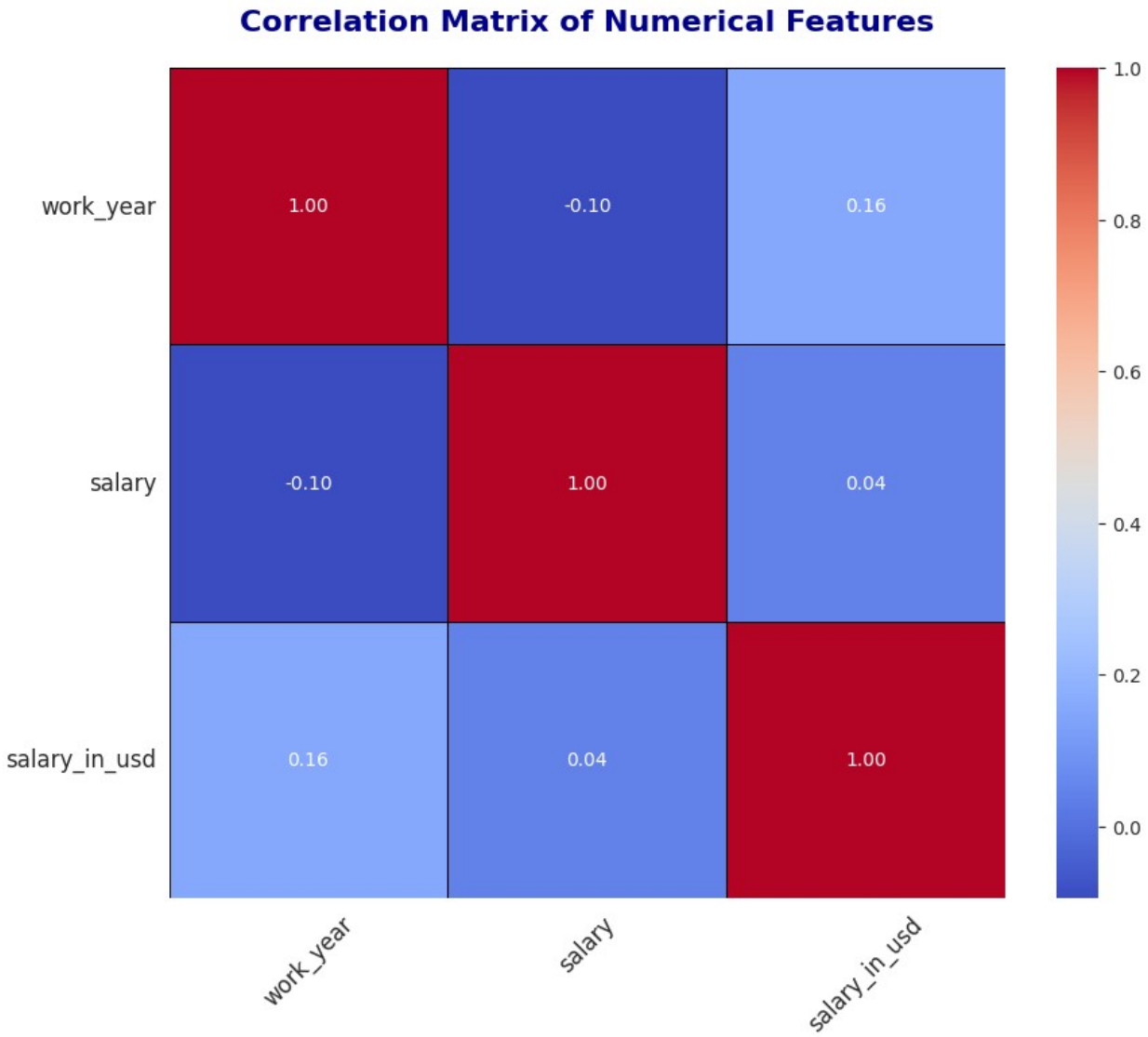

```
sns.set_style("whitegrid")

plt.figure(figsize=(12, 8))
top_10_locations = df.groupby('company_location')
['salary_in_usd'].mean().sort_values(ascending=False).head(10)

# Use a colorful palette
colors = sns.color_palette("Set2", len(top_10_locations))

top_10_locations.plot(kind='bar', color=colors, edgecolor='black')

# Add title and labels
plt.title('Top 10 Locations by Average Salary (USD)', fontsize=16,
fontweight='bold', color='darkblue', pad=20)
plt.xlabel('Company Location', fontsize=14, fontweight='bold',
color='darkgreen')
plt.ylabel('Average Salary (USD)', fontsize=14, fontweight='bold',
color='darkgreen')

# Add gridlines
plt.grid(axis='y', linestyle='--', alpha=0.6)

# Rotate x-axis labels for better readability
plt.xticks(rotation=45, fontsize=12)
plt.yticks(fontsize=12)
```

```
# Remove top and right spines
sns.despine()

# Show the plot
plt.show()
```



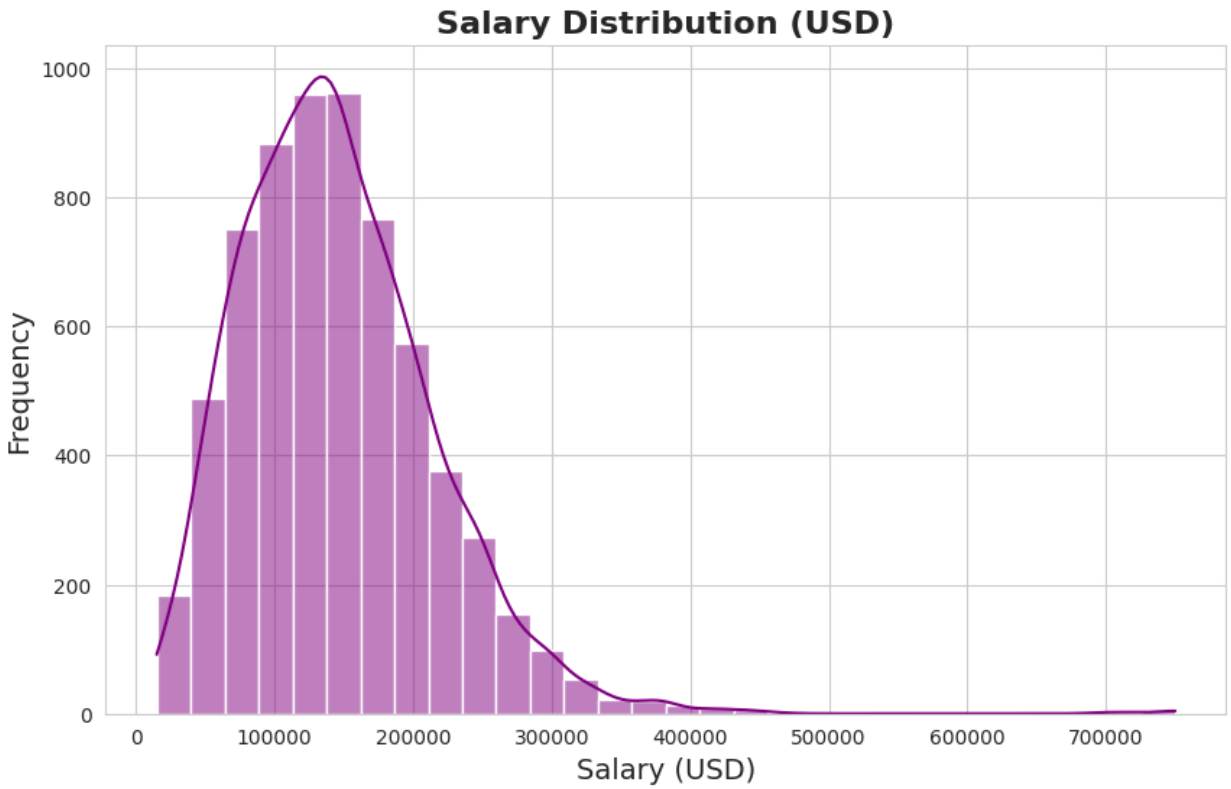

```
# Select only the numerical columns from the DataFrame
numerical_columns = df.select_dtypes(include=['float64', 'int64'])

# Set a larger figure size for better clarity
plt.figure(figsize=(10, 8))

# Create a correlation heatmap
corr_matrix = numerical_columns.corr()
sns.heatmap(corr_matrix, annot=True, fmt='.2f', cmap='coolwarm',
linewidths=0.5, linecolor='black')
```

```
# Add title
plt.title('Correlation Matrix of Numerical Features', fontsize=16,
fontweight='bold', color='darkblue', pad=20)

# Adjust the ticks
plt.xticks(fontsize=12, rotation=45)
plt.yticks(fontsize=12, rotation=0)

# Show the plot
plt.show()
```

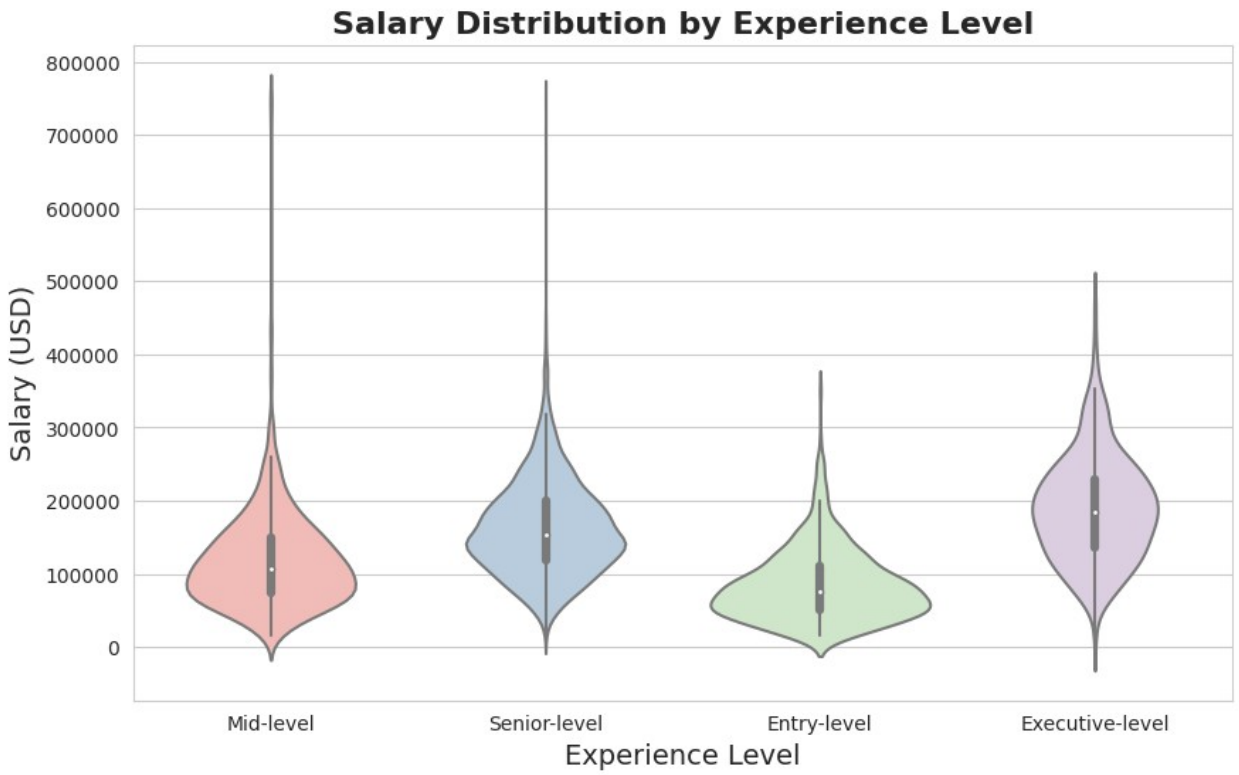



```
plt.figure(figsize=(10, 6))
sns.histplot(df['salary_in_usd'], bins=30, kde=True, color='purple')
plt.title('Salary Distribution (USD)', fontsize=16, fontweight='bold')
plt.xlabel('Salary (USD)', fontsize=14)
```

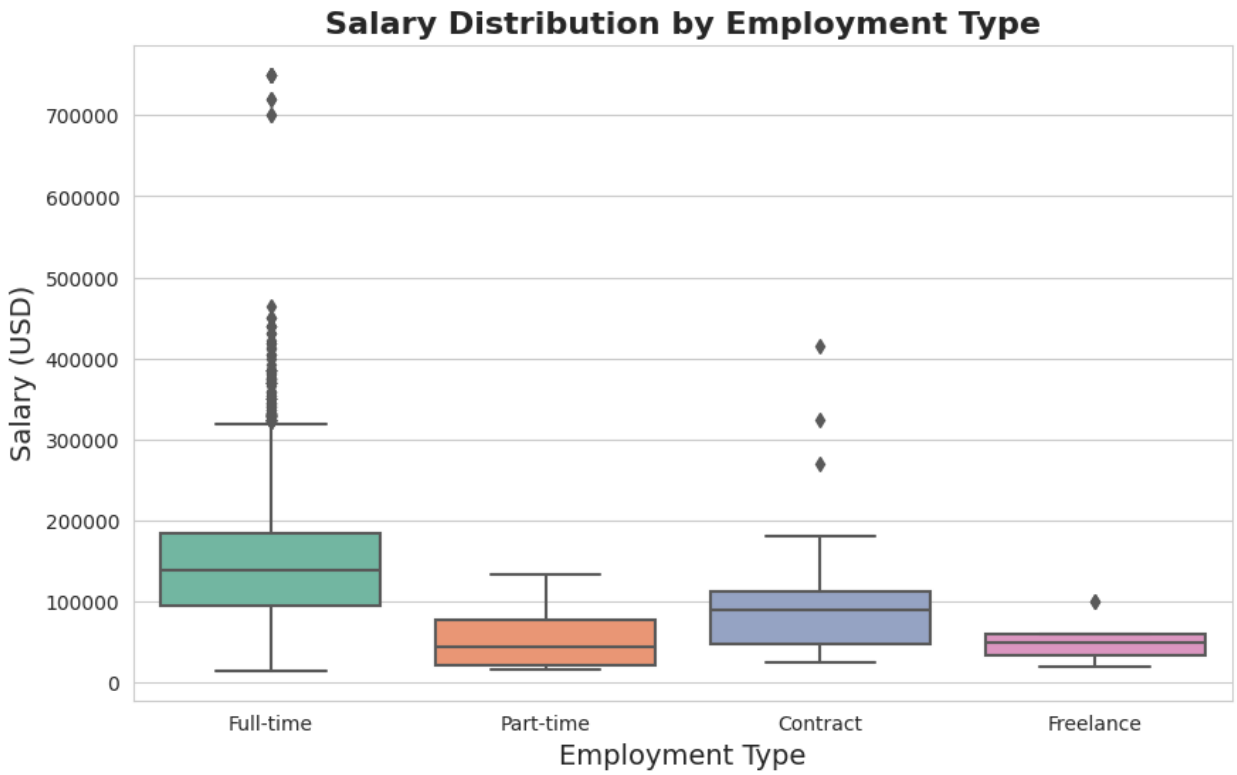
```
plt.ylabel('Frequency', fontsize=14)
plt.show()
```



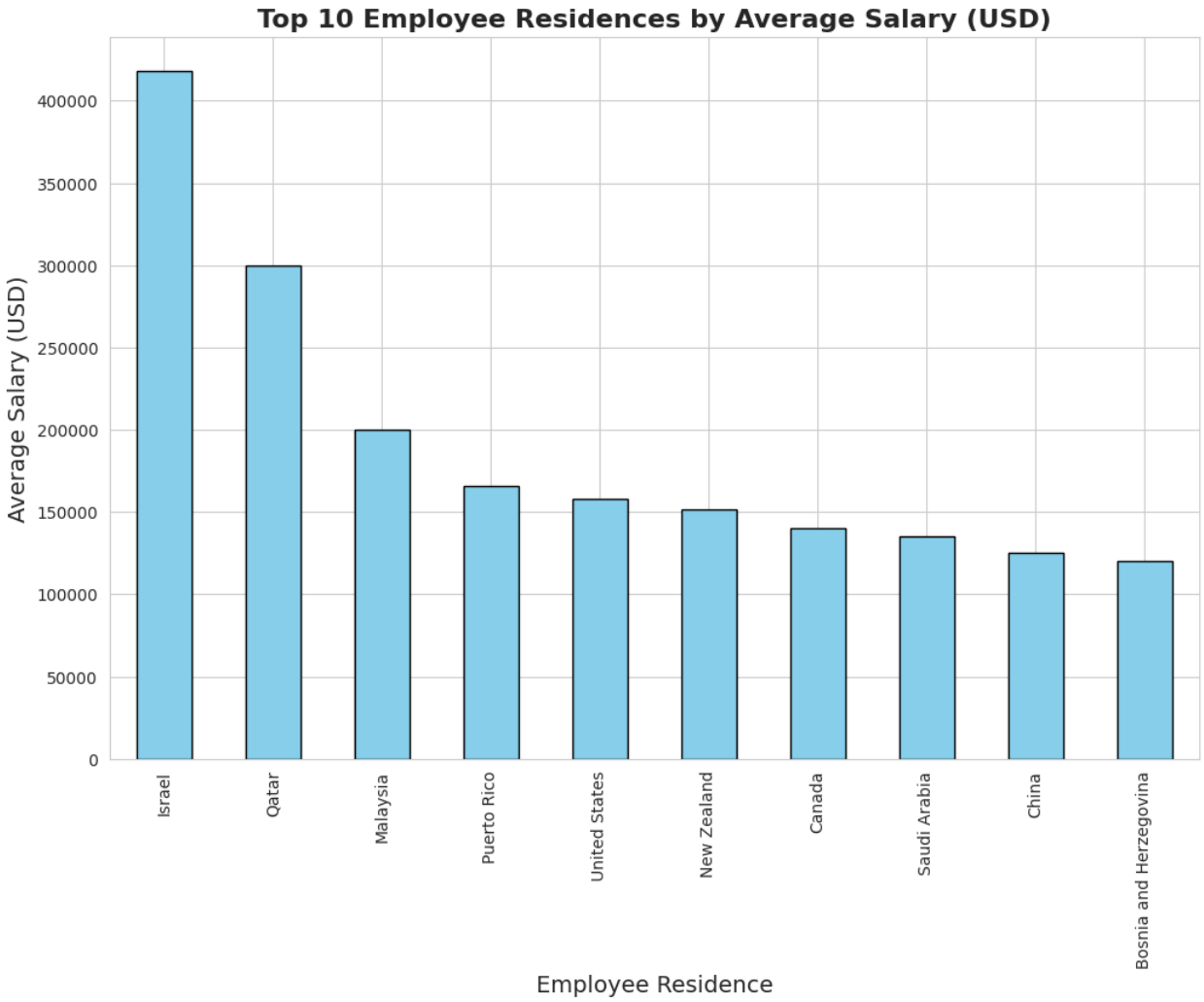

```
plt.figure(figsize=(10, 6))
sns.violinplot(x='experience_level', y='salary_in_usd', data=df,
palette='Pastel1')
plt.title('Salary Distribution by Experience Level', fontsize=16,
fontweight='bold')
plt.xlabel('Experience Level', fontsize=14)
plt.ylabel('Salary (USD)', fontsize=14)
plt.show()
```



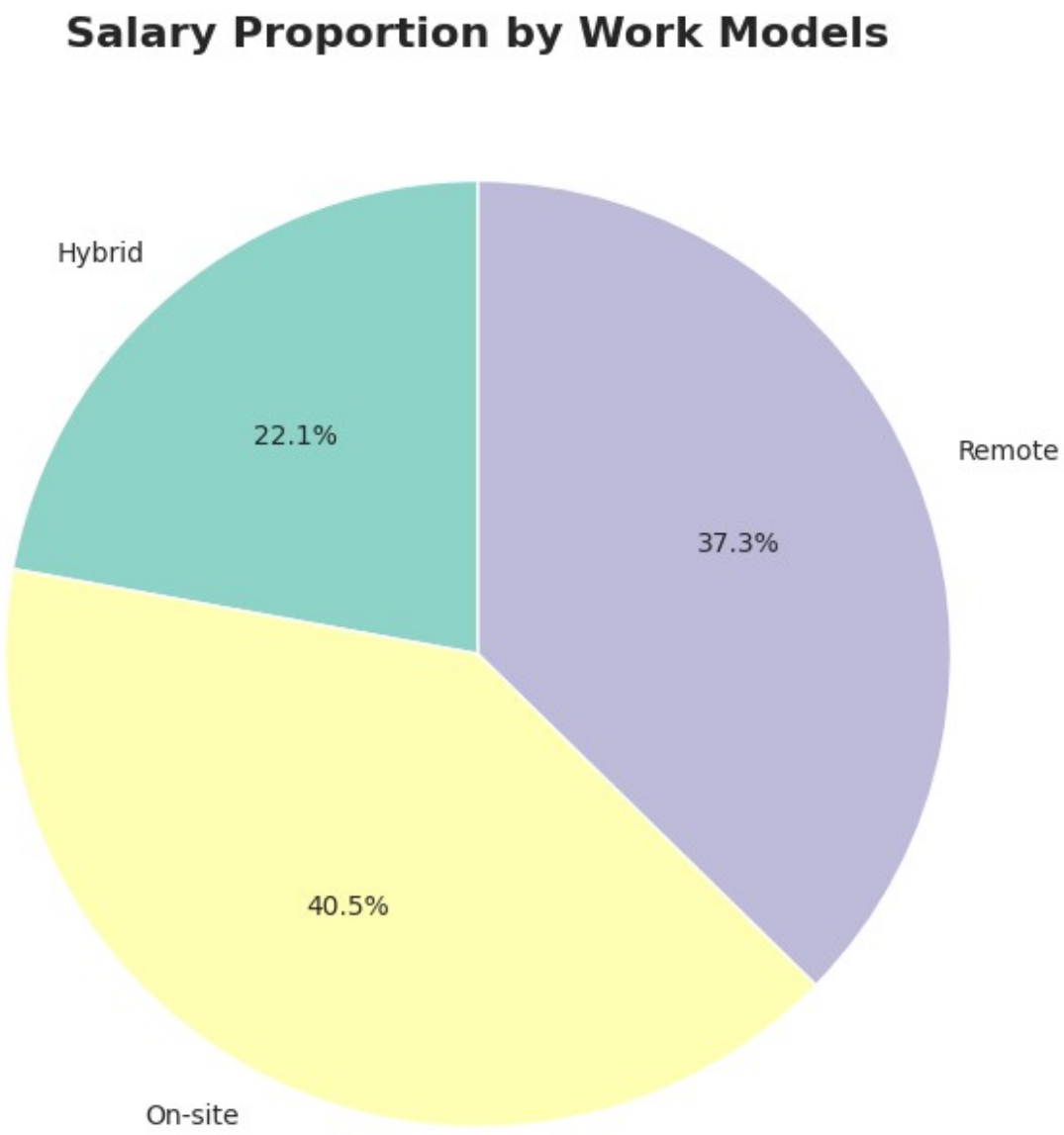

```
plt.figure(figsize=(10, 6))
sns.boxplot(x='employment_type', y='salary_in_usd', data=df,
palette='Set2')
plt.title('Salary Distribution by Employment Type', fontsize=16,
fontweight='bold')
plt.xlabel('Employment Type', fontsize=14)
plt.ylabel('Salary (USD)', fontsize=14)
plt.show()
```



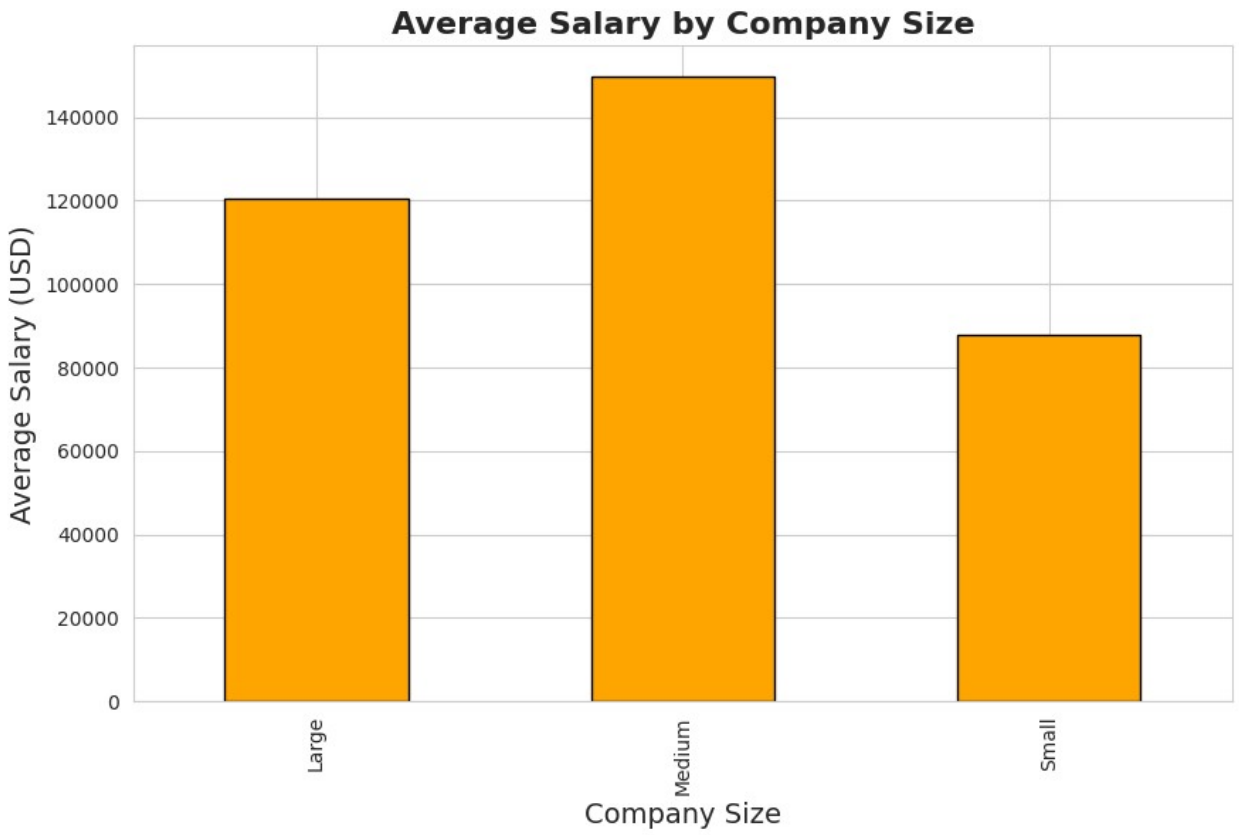

```
plt.figure(figsize=(12, 8))
df.groupby('employee_residence')
['salary_in_usd'].mean().sort_values(ascending=False).head(10).plot(ki
nd='bar', color='skyblue', edgecolor='black')
plt.title('Top 10 Employee Residences by Average Salary (USD)',
fontsize=16, fontweight='bold')
plt.xlabel('Employee Residence', fontsize=14)
plt.ylabel('Average Salary (USD)', fontsize=14)
plt.show()
```




```
plt.figure(figsize=(8, 8))
df.groupby('work_models')['salary_in_usd'].mean().plot(kind='pie',
autopct='%1.1f%%', startangle=90, colors=sns.color_palette('Set3'))
plt.title('Salary Proportion by Work Models', fontsize=16,
fontweight='bold')
plt.ylabel('')
plt.show()
```

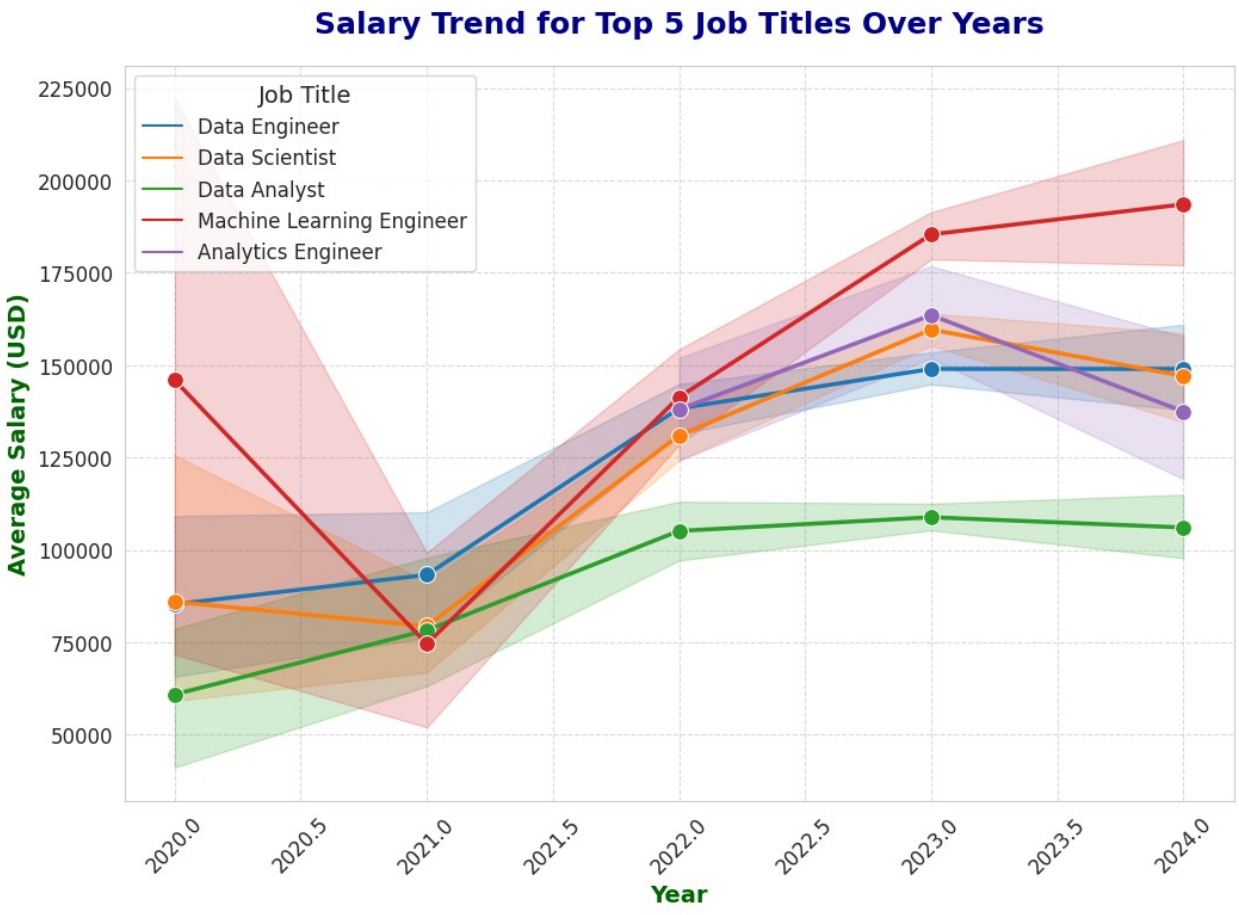



```
plt.figure(figsize=(10, 6))
df.groupby('company_size')['salary_in_usd'].mean().plot(kind='bar',
color='orange', edgecolor='black')
plt.title('Average Salary by Company Size', fontsize=16,
fontweight='bold')
plt.xlabel('Company Size', fontsize=14)
plt.ylabel('Average Salary (USD)', fontsize=14)
plt.show()
```



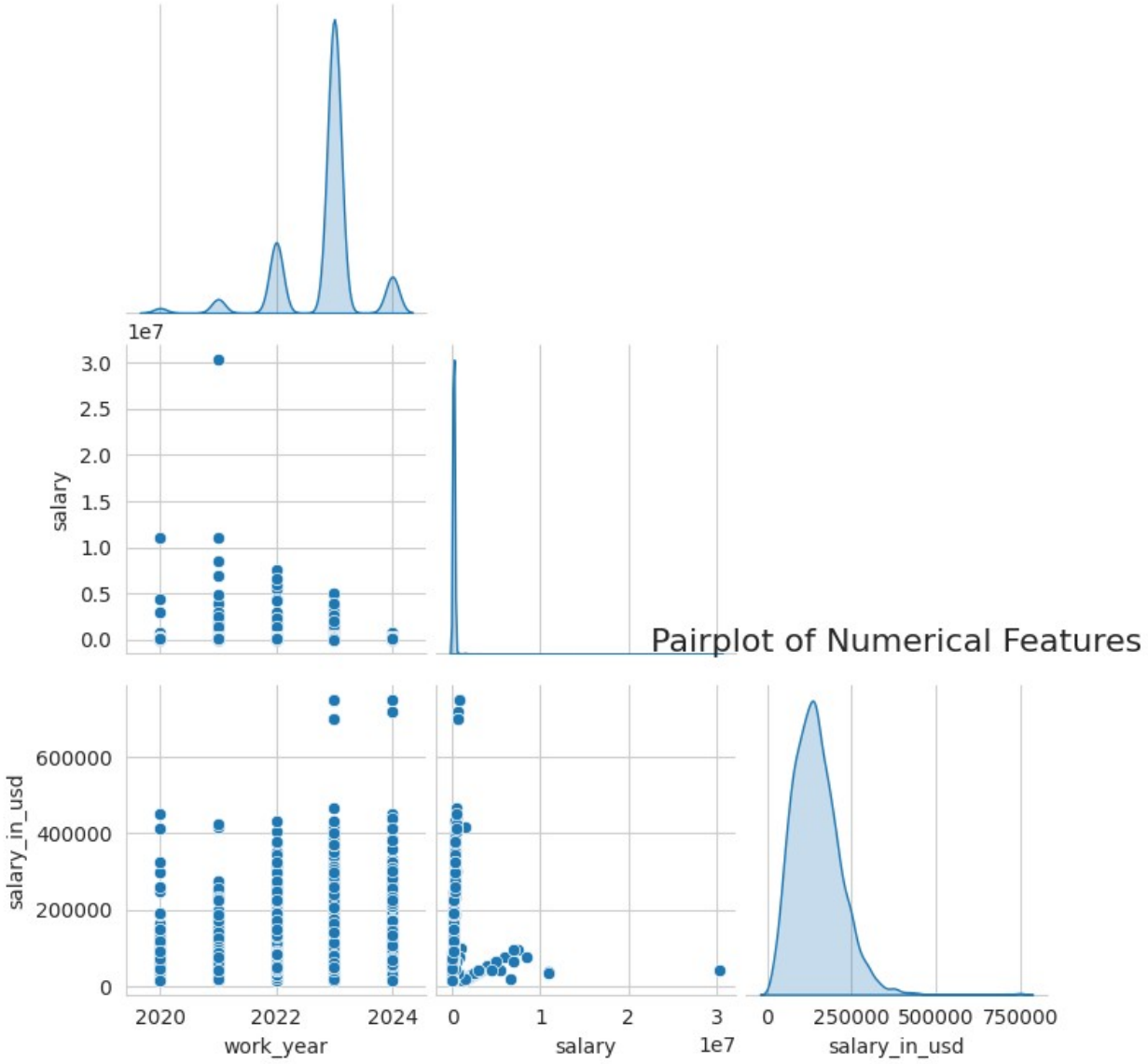

```
# Get the top 5 job titles by frequency
top_5_titles = df['job_title'].value_counts().head(5).index

# Filter the dataset for only the top 5 job titles
df_top_5 = df[df['job_title'].isin(top_5_titles)]

# Create the figure
plt.figure(figsize=(12, 8))

# Plot the line chart
sns.lineplot(
    data=df_top_5,
    x='work_year',
    y='salary_in_usd',
    hue='job_title',
    marker='o',
    markersize=10,  # Larger markers
    linewidth=2.5   # Thicker lines
)

# Add a title and labels with improved font size and weight
plt.title('Salary Trend for Top 5 Job Titles Over Years', fontsize=18,
fontweight='bold', color='darkblue', pad=20)
plt.xlabel('Year', fontsize=14, fontweight='bold', color='darkgreen')
```

```
plt.ylabel('Average Salary (USD)', fontsize=14, fontweight='bold',
color='darkgreen')

# Customize the legend for better readability
plt.legend(title='Job Title', fontsize=12, title_fontsize=14)

# Show gridlines for better clarity
plt.grid(True, linestyle='--', alpha=0.7)

# Adjust the x-ticks and y-ticks for better readability
plt.xticks(fontsize=12, rotation=45)
plt.yticks(fontsize=12)

# Display the plot
plt.show()
```



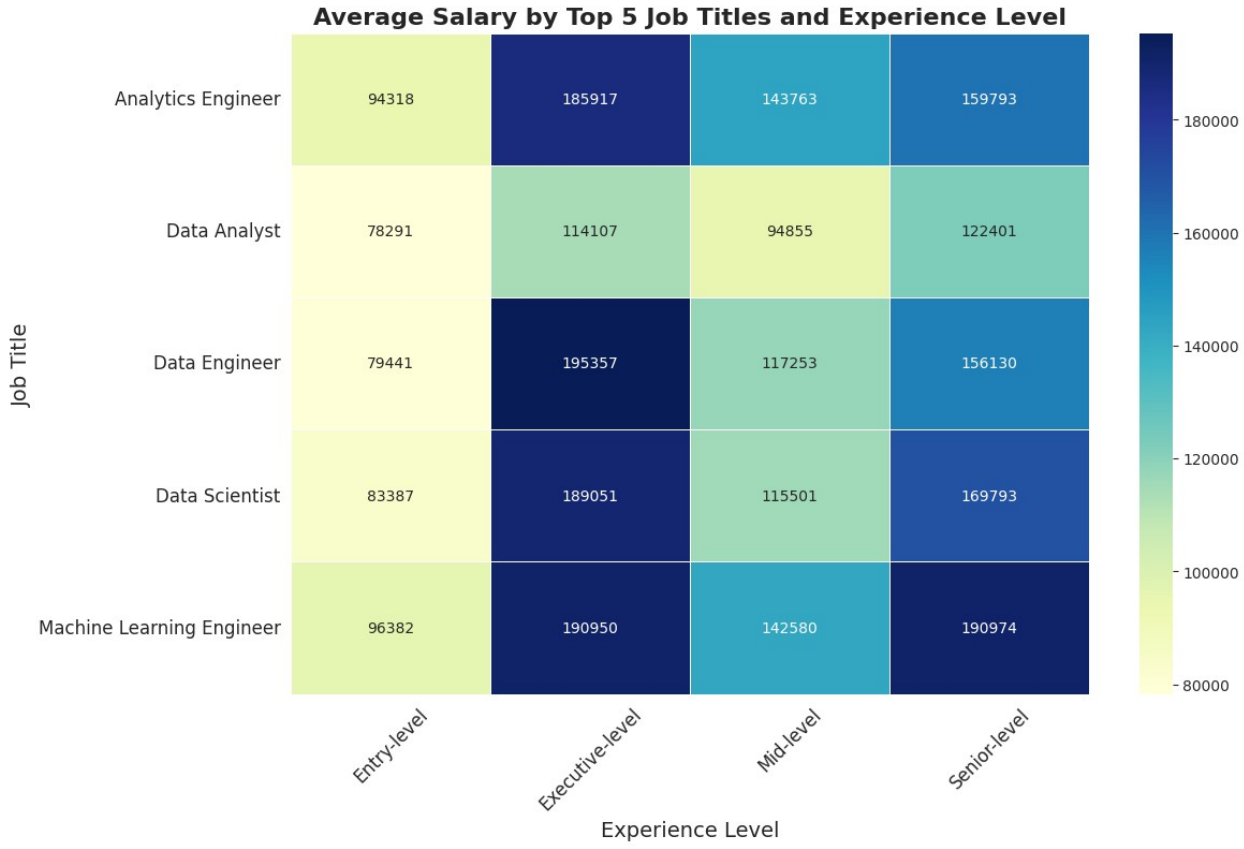

```
plt.figure(figsize=(10, 10))
sns.pairplot(df.select_dtypes(include=['float64', 'int64']),
corner=True, diag_kind='kde', palette='coolwarm')
plt.title('Pairplot of Numerical Features', fontsize=16)
plt.show()
```

```
<Figure size 1000x1000 with 0 Axes>
```



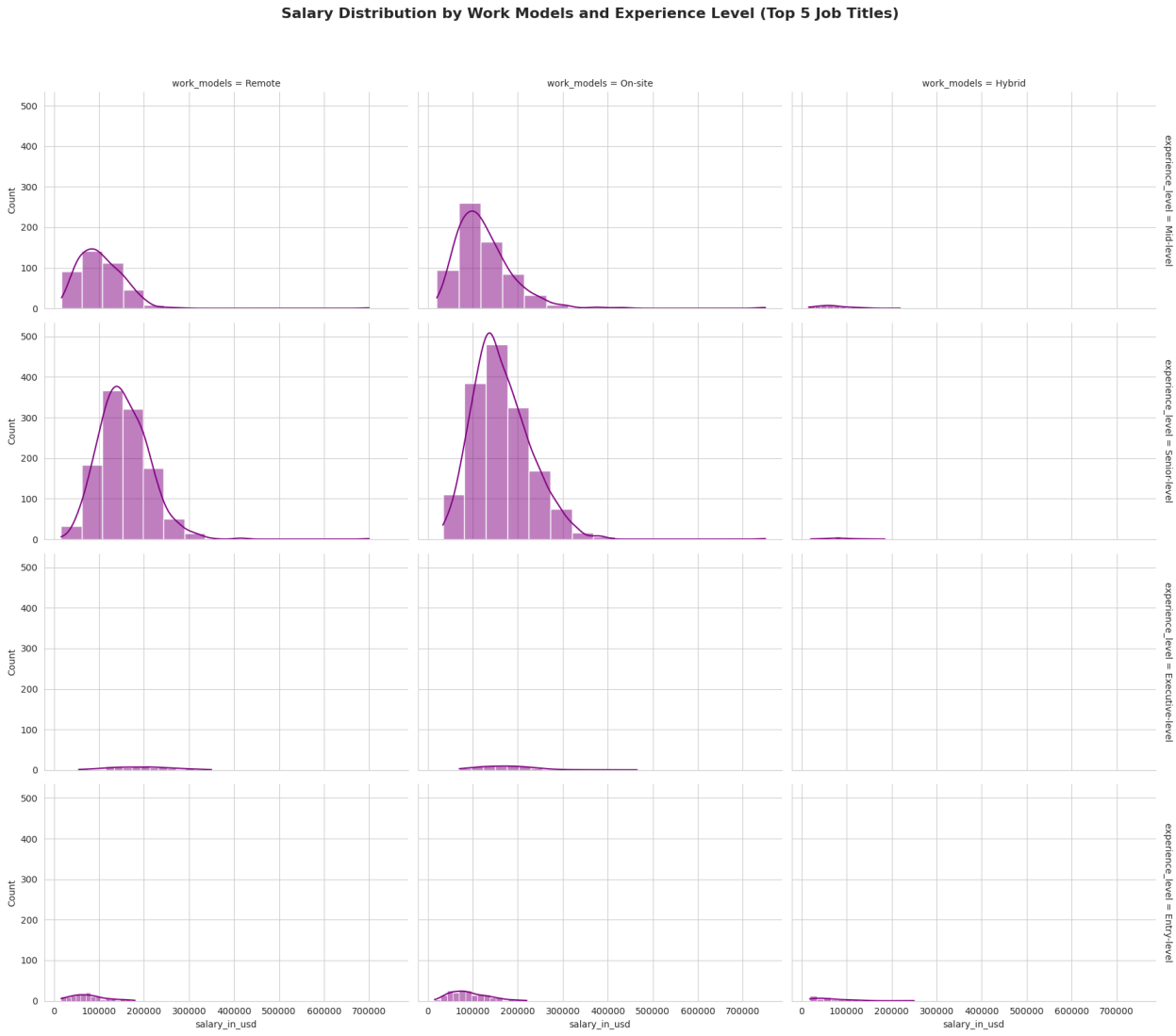

```
top_5_titles = df['job_title'].value_counts().head(5).index
df_top_5 = df[df['job_title'].isin(top_5_titles)]

pivot_table = df_top_5.pivot_table(values='salary_in_usd',
index='job_title', columns='experience_level', aggfunc='mean')

plt.figure(figsize=(12, 8))
sns.heatmap(pivot_table, annot=True, fmt='.0f', cmap='YlGnBu',
linewidths=0.5)

plt.title('Average Salary by Top 5 Job Titles and Experience Level',
fontsize=16, fontweight='bold')
```

```
plt.xlabel('Experience Level', fontsize=14)
plt.ylabel('Job Title', fontsize=14)
plt.xticks(rotation=45, fontsize=12)
plt.yticks(fontsize=12)
plt.show()
```



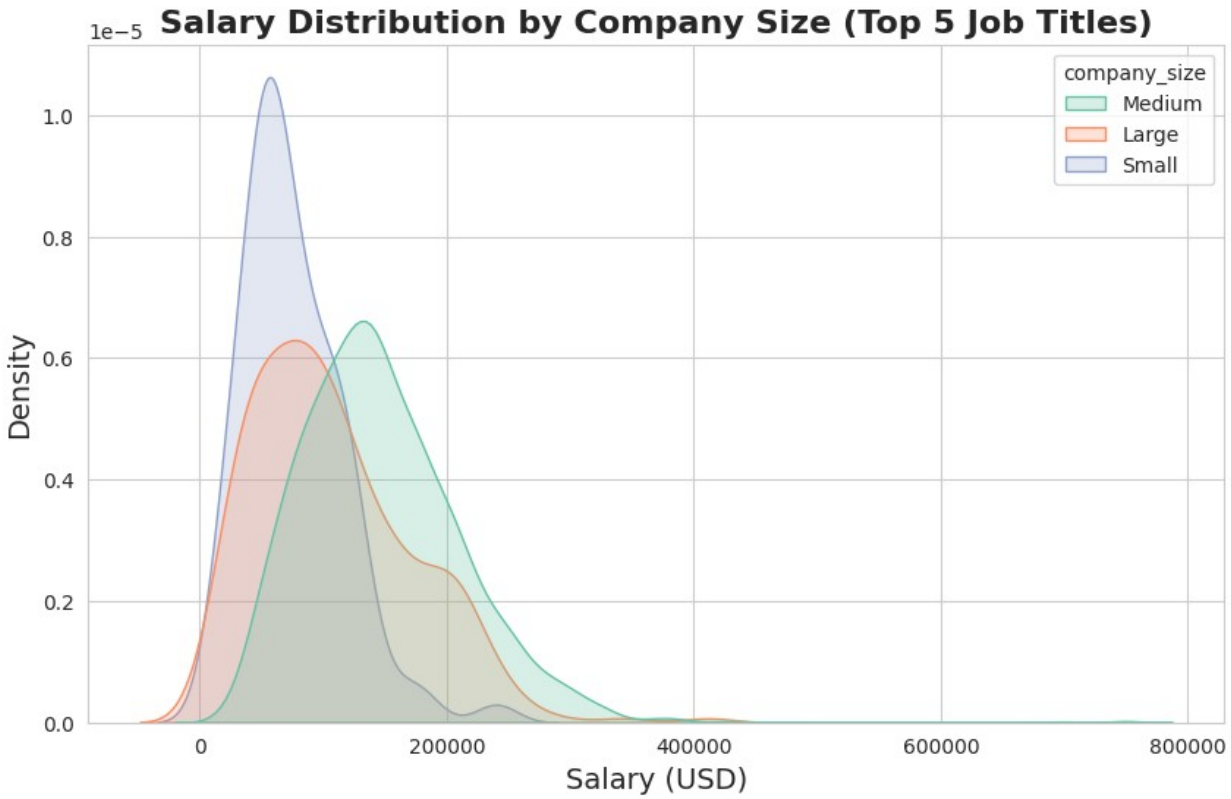

```
# Filter out combinations with less than 10 data points for better
visibility
df_filtered = df_top_5.groupby(['work_models',
'experience_level']).filter(lambda x: len(x) > 10)

# Create the FacetGrid
g = sns.FacetGrid(df_filtered, col="work_models",
row="experience_level", margin_titles=True, height=4, aspect=1.5)

# Adjust the number of bins based on the data range
g.map(sns.histplot, "salary_in_usd", bins=15, color="purple",
kde=True)

# Add title and adjust the layout
g.fig.subplots_adjust(top=0.9)
g.fig.suptitle('Salary Distribution by Work Models and Experience
Level (Top 5 Job Titles)', fontsize=16, fontweight='bold')
```

```
plt.show()
```




```
plt.figure(figsize=(12, 8))
sns.swarmplot(x='job_title', y='salary_in_usd',
hue='experience_level', data=df_top_5, palette='coolwarm', size=7)

plt.title('Salary by Job Title and Experience Level (Top 5 Job
Titles)', fontsize=16, fontweight='bold')
plt.xlabel('Job Title', fontsize=14)
plt.ylabel('Salary (USD)', fontsize=14)
plt.xticks(rotation=45)
plt.legend(title='Experience Level', fontsize=12)
plt.show()
```


```
plt.figure(figsize=(10, 6))
sns.kdeplot(data=df_top_5, x='salary_in_usd', hue='company_size',
fill=True, palette='Set2', common_norm=False)

plt.title('Salary Distribution by Company Size (Top 5 Job Titles)',
fontsize=16, fontweight='bold')
plt.xlabel('Salary (USD)', fontsize=14)
plt.ylabel('Density', fontsize=14)
plt.show()
```




```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error

df.columns

Index(['job_title', 'experience_level', 'employment_type',
'work_models',
       'work_year', 'employee_residence', 'salary_in_usd',
'company_location',
       'company_size'],
      dtype='object')

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

X = pd.get_dummies(X)

from sklearn.model_selection import train_test_split

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

from sklearn.linear_model import LinearRegression
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor,
```

```
GradientBoostingRegressor
from sklearn.svm import SVR
from sklearn.neighbors import KNeighborsRegressor
from sklearn.neural_network import MLPRegressor
from sklearn.metrics import mean_squared_error

# Initialize models
linear_model = LinearRegression()
decision_tree_model = DecisionTreeRegressor()
random_forest_model = RandomForestRegressor()
gradient_boosting_model = GradientBoostingRegressor()
svr_model = SVR()
knn_model = KNeighborsRegressor()
neural_network_model = MLPRegressor(max_iter=1000)

# Train models
linear_model.fit(X_train, y_train)
decision_tree_model.fit(X_train, y_train)
random_forest_model.fit(X_train, y_train)
gradient_boosting_model.fit(X_train, y_train)
svr_model.fit(X_train, y_train)
knn_model.fit(X_train, y_train)
neural_network_model.fit(X_train, y_train)

# Predict
linear_predictions = linear_model.predict(X_test)
decision_tree_predictions = decision_tree_model.predict(X_test)
random_forest_predictions = random_forest_model.predict(X_test)
gradient_boosting_predictions =
gradient_boosting_model.predict(X_test)
svr_predictions = svr_model.predict(X_test)
knn_predictions = knn_model.predict(X_test)
neural_network_predictions = neural_network_model.predict(X_test)

# Evaluate
print("Linear Regression MSE:", mean_squared_error(y_test,
linear_predictions))
print("Decision Tree MSE:", mean_squared_error(y_test,
decision_tree_predictions))
print("Random Forest MSE:", mean_squared_error(y_test,
random_forest_predictions))
print("Gradient Boosting MSE:", mean_squared_error(y_test,
gradient_boosting_predictions))
print("SVR MSE:", mean_squared_error(y_test, svr_predictions))
print("KNN MSE:", mean_squared_error(y_test, knn_predictions))
print("Neural Network MSE:", mean_squared_error(y_test,
neural_network_predictions))

Linear Regression MSE: 3.3553411212279144e+33
Decision Tree MSE: 4277056006.2234654
```

```
Random Forest MSE: 4057114571.330849
Gradient Boosting MSE: 4072153298.6079555
SVR MSE: 5920428196.033797
KNN MSE: 4582486883.346151
Neural Network MSE: 5455416831.58686
```

```
from sklearn.metrics import mean_squared_error, r2_score

linear_mse = mean_squared_error(y_test, linear_predictions)
decision_tree_mse = mean_squared_error(y_test,
decision_tree_predictions)
random_forest_mse = mean_squared_error(y_test,
random_forest_predictions)
gradient_boosting_mse = mean_squared_error(y_test,
gradient_boosting_predictions)
svr_mse = mean_squared_error(y_test, svr_predictions)
knn_mse = mean_squared_error(y_test, knn_predictions)
neural_network_mse = mean_squared_error(y_test,
neural_network_predictions)

linear_r2 = r2_score(y_test, linear_predictions)
decision_tree_r2 = r2_score(y_test, decision_tree_predictions)
random_forest_r2 = r2_score(y_test, random_forest_predictions)
gradient_boosting_r2 = r2_score(y_test, gradient_boosting_predictions)
svr_r2 = r2_score(y_test, svr_predictions)
knn_r2 = r2_score(y_test, knn_predictions)
neural_network_r2 = r2_score(y_test, neural_network_predictions)

print("Linear Regression MSE:", linear_mse)
print("Linear Regression R-squared:", linear_r2)
print("Decision Tree MSE:", decision_tree_mse)
print("Decision Tree R-squared:", decision_tree_r2)
print("Random Forest MSE:", random_forest_mse)
print("Random Forest R-squared:", random_forest_r2)
print("Gradient Boosting MSE:", gradient_boosting_mse)
print("Gradient Boosting R-squared:", gradient_boosting_r2)
print("SVR MSE:", svr_mse)
print("SVR R-squared:", svr_r2)
print("KNN MSE:", knn_mse)
print("KNN R-squared:", knn_r2)
print("Neural Network MSE:", neural_network_mse)
print("Neural Network R-squared:", neural_network_r2)
```

```
Linear Regression MSE: 3.3553411212279144e+33
Linear Regression R-squared: -5.7092746237210695e+23
Decision Tree MSE: 4277056006.2234654
Decision Tree R-squared: 0.27223830787049375
Random Forest MSE: 4057114571.330849
Random Forest R-squared: 0.3096624029943207
Gradient Boosting MSE: 4072153298.6079555
```

```
Gradient Boosting R-squared: 0.3071034910710876
SVR MSE: 5920428196.033797
SVR R-squared: -0.007389390227142911
KNN MSE: 4582486883.346151
KNN R-squared: 0.22026777214686322
Neural Network MSE: 5455416831.58686
Neural Network R-squared: 0.07173453449044986

###***Summary of findings***###
```

World data:

Average salary of a DS is ~155k compared to a DA of ~108k. There's around a $50k difference of salaries in each percentile. Average salary of an entry level DS is ~85k compared to ~78k of a DA. Overall, for an entry level job, DS and DA can expect a similar salary, even across percentiles. Some of the best countries in terms of USD salary are: United Sates, Canada, New Zealand, and Australia. There's a visually positive correlation of experience level and salary. However, location and total living costs should be taken into consideration. United States data:

In the United States, average salary of a DS is ~166k compared to ~112k of a DA. In the United States, average salary of an entry level DS is ~104k compared to ~85k of a DA. Senior level DS can expect an average salary closer to 200k, meanwhile DA can expect an average around 125k. Conclusions:

United States has the highest average salaries of both Data Scientists and Data Analysts. Also, their average salary and entry level salary is also higher compared to the average of the rest of the world. In terms of salary, it's the best destination.

For Data Scientists, there's a slight advantage in becoming executive-level since the salaries have less variance and have a slight higher average. However, the amount of executive-level jobs is significantly less and senior-level has more salaries > 300k.

Mid-sized companies seem to be the sweet spot between salary and job security. However, it should be taken with a grain of salt since most of the salaries came from mid-sized companies.

```

```