Stack Overflow Annual Developer Survey - 2023

Stack overflow, the arguably largest platform to discuss and learn coding has been hosting a survey for developers since 2011 now. The survey covers a wide range of topics, including programming languages, frameworks, tools, technologies, job satisfaction, and demographics. It is one of the largest surveys of its kind and provides valuable information about the preferences, trends, and challenges in the software development industry.

The present data analysis project builds upon the 2023 survey and explores some of its features with emphasis on the data and business analysis field of industry. The survey data has already been cleaned by the stack overflow team and is ready to be used for anaylsis. As the survey result csv file is too large for githubs default upload I'll reference it here instead.
You can find the file at insights.stackoverflow.com/survey

In [ ]:
```
# import necessary libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from IPython.display import HTML, display
```

In [ ]:
```
# load csv file
df = pd.read_csv("stack-overflow-developer-survey-2023/survey_results_pub

# List the column names
display(df.columns)
```

```
Index(['ResponseId', 'Q120', 'MainBranch', 'Age', 'Employment', 'RemoteWor
k',
       'CodingActivities', 'EdLevel', 'LearnCode', 'LearnCodeOnline',
       'LearnCodeCoursesCert', 'YearsCode', 'YearsCodePro', 'DevType',
       'OrgSize', 'PurchaseInfluence', 'TechList', 'BuyNewTool', 'Countr
y',
       'Currency', 'CompTotal', 'LanguageHaveWorkedWith',
       'LanguageWantToWorkWith', 'DatabaseHaveWorkedWith',
       'DatabaseWantToWorkWith', 'PlatformHaveWorkedWith',
       'PlatformWantToWorkWith', 'WebframeHaveWorkedWith',
       'WebframeWantToWorkWith', 'MiscTechHaveWorkedWith',
       'MiscTechWantToWorkWith', 'ToolsTechHaveWorkedWith',
       'ToolsTechWantToWorkWith', 'NEWCollabToolsHaveWorkedWith',
       'NEWCollabToolsWantToWorkWith', 'OpSysPersonal use',
       'OpSysProfessional use', 'OfficeStackAsyncHaveWorkedWith',
       'OfficeStackAsyncWantToWorkWith', 'OfficeStackSyncHaveWorkedWith',
       'OfficeStackSyncWantToWorkWith', 'AISearchHaveWorkedWith',
       'AISearchWantToWorkWith', 'AIDevHaveWorkedWith', 'AIDevWantToWorkWi
th',
       'NEWSOSites', 'SOVisitFreq', 'SOAccount', 'SOPartFreq', 'SOComm',
       'SOAI', 'AISelect', 'AISent', 'AIAcc', 'AIBen',
       'AIToolInterested in Using', 'AIToolCurrently Using',
       'AIToolNot interested in Using', 'AINextVery different',
       'AINextNeither different nor similar', 'AINextSomewhat similar',
       'AINextVery similar', 'AINextSomewhat different', 'TBranch', 'ICorP
M',
       'WorkExp', 'Knowledge_1', 'Knowledge_2', 'Knowledge_3', 'Knowledge_
4',
       'Knowledge_5', 'Knowledge_6', 'Knowledge_7', 'Knowledge_8',
       'Frequency_1', 'Frequency_2', 'Frequency_3', 'TimeSearching',
       'TimeAnswering', 'ProfessionalTech', 'Industry', 'SurveyLength',
       'SurveyEase', 'ConvertedCompYearly'],
      dtype='object')
```

Schema

How do the questions from the survey translate to column names?

In [ ]:
```
# Load the schema file
schema = pd.read_csv("stack-overflow-developer-survey-2023/survey_results

# To prevent truncation of columns, change the default options
pd.set_option('display.max_colwidth', None)

# To prevent truncation of cells, change the default options
pd.set_option('display.max_rows', None)

# Truncate introductory text
schema_display = schema[["qname", "question"]][4:]

# Define the styles for the display, Set the alignment for the 'question'
styles = [
    {'selector': 'td', 'props': [('text-align', 'left')]}
]

# Apply styles to the DataFrame
schema_display.style.set_table_styles(styles)
```

Out[ ]:

	qname	question
4	S1	**Basic Information** The first section will focus on some basic information about who you are. Most questions in this section are required. Required questions are noted with *.
5	MainBranch	Which of the following options best describes you today? For the purpose of this survey, a developer is "someone who writes code". *
6	Age	What is your age? *
7	Employment	Which of the following best describes your current employment status? Select all that apply.
8	RemoteWork	Which best describes your current work situation?
9	CodingActivities	Which of the following best describes the code you write outside of work? Select all that apply.
10	S2	**Education, work, and career** This section will focus on your education, work, and career. Most questions in this section are optional. Required questions are noted with *.
11	EdLevel	Which of the following best describes the highest level of formal education that you've completed? *
12	LearnCode	How do you learn to code? Select all that apply.
13	LearnCodeOnline	What online resources do you use to learn to code? Select all that apply.
14	LearnCodeCoursesCert	What online courses or certifications do you use to learn to code? Select all that apply.
15	YearsCode	Including any education, how many years have you been coding in total?
16	YearsCodePro	NOT including education, how many years have you coded professionally (as a part of your work)?
17	DevType	Which of the following describes your current job, the one you do most of the time? Please select only one.
18	OrgSize	Approximately how many people are employed by the company or organization you currently work for? This should only include your primary company, and not the entire holding or parent company if that applies.

	qname	question
19	PurchaseInfluence	What level of influence do you, personally, have over new technology purchases at your organization?
20	TechList	When thinking about new technology purchases at your organization, are you more likely to be given a short list of products/services to evaluate or be told to investigate on your own?
21	BuyNewTool	When buying a new tool or software, how do you discover and research available solutions? Select all that apply.
22	Country	Where do you live? *****
23	Currency	Which currency do you use day-to-day? If your answer is complicated, please pick the one you're most comfortable estimating in. *
24	CompTotal	What is your current total **annual** compensation (salary, bonuses, and perks, before taxes and deductions)? Please enter a whole number in the box below, without any punctuation. If you are paid hourly, please estimate an equivalent yearly salary. If you prefer not to answer, please leave the box empty.
25	S3	**Tech and tech culture** The next set of questions will focus on technology and tech culture. Most questions in this section are optional. Required questions are noted with *.
26	Language	Which **programming, scripting, and markup languages** have you done extensive development work in over the past year, and which do you want to work in over the next year? (If you both worked with the language and want to continue to do so, please check both boxes in that row.)
27	Database	Which **database environments** have you done extensive development work in over the past year, and which do you want to work in over the next year? (If you both worked with the database and want to continue to do so, please check both boxes in that row.)
28	Platform	Which **cloud platforms** have you done extensive development work in over the past year, and which do you want to work in over the next year? (If you both worked with the platform and want to continue to do so, please check both boxes in that row.)
29	Webframe	Which **web frameworks and web technologies** have you done extensive development work in over the past year, and which do you want to work in over the next year? (If you both worked with the framework and want to continue to do so, please check both boxes in that row.)

	qname	question
30	MiscTech	Which **other frameworks and libraries** have you done extensive development work in over the past year, and which do you want to work in over the next year? (If you both worked with the framework and want to continue to do so, please check both boxes in that row.)
31	ToolsTech	Which **developer tools for compiling, building and testing** have you done extensive development work in over the past year, and which do you want to work in over the next year? (If you both worked with the technology and want to continue to do so, please check both boxes in that row.)
32	NEWCollabTools	Which **development environments** did you use regularly over the past year, and which do you want to work with over the next year? Please check all that apply.
33	OpSys	What is the primary **operating system** in which you work?
34	OfficeStackAsync	Which **collaborative work management and/or code documentation tools** did you use regularly over the past year, and which do you want to work with over the next year? Select all that apply
35	OfficeStackSync	Which **communication tools** did you use regularly over the past year, and which do you want to work with over the next year? Select all that apply
36	AISearch	Which **AI-powered search tools** did you use regularly over the past year, and which do you want to work with over the next year? Select all that apply.
37	AIDev	Which **AI-powered developer tools** did you use regularly over the past year, and which do you want to work with over the next year? Select all that apply
38	S4	# Stack Overflow - usage and community This section will focus on Stack Overflow usage and community questions. Most questions in this section are optional. Required questions are noted with *.
39	NEWSOSites	Which of the following Stack Overflow sites have you visited? Select all that apply. *
40	SOVisitFreq	How frequently would you say you visit Stack Overflow?
41	SOAccount	Do you have a Stack Overflow account?
42	SOPartFreq	How frequently would you say you participate in Q&A on Stack Overflow? By participate we mean ask, answer, vote for, or comment on questions.

	qname	question
43	SOComm	Do you consider yourself a member of the Stack Overflow community?
44	S5	# Artificial Intelligence (AI) This section will focus on AI usage and sentiment. Most questions in this section are optional. Required questions are noted with *.
45	SOAI	Artificial Intelligence (AI) tools have gained prominence recently across industries. The following question asks for your perspective on Stack Overflow using AI technology to improve the current experience, for example by using AI to suggest better titles for your question. For your response, consider these sorts of improvements to the experience with/on Stack Overflow and not the presence or absence of content generated by AI and posted on Stack Overflow. What is your opinion on Stack Overflow using AI tools to improve the current experience? What could AI help with most to improve Stack Overflow?
46	AISelect	Do you currently use AI tools in your development process? *
47	AISent	How favorable is your stance on using AI tools as part of your development workflow?
48	AIAcc	For the AI tools you use as part of your development workflow, what are the MOST important benefits you are hoping to achieve? Please check all that apply.
49	AIBen	How much do you trust the accuracy of the output from AI tools as part of your development workflow?
50	AITool	Which parts of your development workflow are you currently using AI tools for and which are you interested in using AI tools for over the next year? Please select all that apply.
51	AINext	Thinking about how your workflow and process changes over time, how similar or different do you anticipate your workflow to be 1 year from now as a result of AI tools you are currently using?
52	AIOpen	Please describe how you would expect your workflow to be different, if at all, in 1 year as a result of AI advancements.
53	S6	# Professional Developer Series In one of our previous pulse surveys, we saw that people valued being productive and having growth opportunities at work. By adding a **Professional Developer Series** to the survey we want to better understand the things that may

	qname	question
		impact your productivity and opportunities to learn. Like the rest of the survey, your responses are completely anonymous. Most questions in this section are optional. Required questions are noted with *. Blog post - New data: What developers look for in future job opportunities - December 7, 2021
54	TBranch	Would you like to participate in the Professional Developer Series? *
55	ICorPM	Are you an individual contributor or people manager?
56	WorkExp	How many years of working experience do you have?
57	Knowledge	Please rate your level of agreement with the following statement:
58	Frequency	How frequently do you experience each of the following?
59	TimeSearching	On an average day, how much time do you typically spend searching for answers or solutions to problems you encounter at work? (This includes time spent searching on your own, asking a colleague, and waiting for a response).
60	TimeAnswering	On an average day, how much time do you typically spend answering questions you get asked at work?
61	ProfessionalTech	My company has:
62	Industry	What industry is the company you work for in?
63	SOTeamsUsage	Does your team use Stack Overflow for Teams?
64	S7	**Final Questions**
65	SurveyLength	How do you feel about the length of the survey this year?
66	SurveyEase	How easy or difficult was this survey to complete?
67	Knowledge_1	I have interactions with people outside of my immediate team.
68	Knowledge_2	Knowledge silos prevent me from getting ideas across the organization (i.e., one individual or team has information that isn't shared with others)
69	Knowledge_3	I can find up-to-date information within my organization to help me do my job.
70	Knowledge_4	I am able to quickly find answers to my questions with existing tools and resources.
71	Knowledge_5	I know which system or resource to use to find information and answers to questions I have.

	qname	question
72	Knowledge_6	I often find myself answering questions that I've already answered before.
73	Knowledge_7	Waiting on answers to questions often causes interruptions and disrupts my workflow.
74	Knowledge_8	I feel like I have the tools and/or resources to quickly understand and work on any area of my company's code/system/platform.
75	Frequency_1	Needing help from people outside of your immediate team?
76	Frequency_2	Interacting with people outside of your immediate team?
77	Frequency_3	Encountering knowledge silos (where one individual or team has information that's not shared or distributed with other individuals or teams) at work?

Whats the top compensation in each industry field?

With the inflation ratio ever rising its interesting to see which industry field yields the highest compensation in 2023.

The listed compensation values aren't all in the same currency. As the survey lists 144 different currencies it would be a huge task to find the exchange rate for each currency to, say EUR.
Instead I chose the 10 most popular currencies listed that amount to a cumulative coverage of 80% of the total dataset.
For these 10 currencies I applied exchange rates to EUR to get a better picture of how the compensations compare to each other

In [ ]:
```
# How many unique currencies does the dataframe list?
display(df["Currency"].nunique()) # 144

# How many of the most popular currencies represent a coverage of 80% of
# Calculate the value counts of each currency
currency_counts = df["Currency"].value_counts()

# Sort the currency counts in descending order
currency_counts_sorted = currency_counts.sort_values(ascending=False)

# Calculate the cumulative sum of the sorted counts
cumulative_sum = currency_counts_sorted.cumsum()

# Find the number of currencies needed for 80% coverage, without NaN entr
threshold = 0.8 * (len(df)- df["Currency"].isna().sum())
num_top_currencies = (cumulative_sum <= threshold).sum()
```

```
# 10 most popular currencies by name and count
top_currencies = df["Currency"].value_counts().head(10)

print(f"Number of top currencies needed for 80% coverage: {num_top_curren

display(top_currencies)
```

```
144
Number of top currencies needed for 80% coverage: 10
Currency
EUR European Euro            17651
USD\tUnited States dollar    16729
GBP\tPound sterling           4473
INR\tIndian rupee             3615
CAD\tCanadian dollar          2647
PLN\tPolish zloty             1606
AUD\tAustralian dollar        1594
BRL\tBrazilian real           1475
SEK\tSwedish krona            1324
CHF\tSwiss franc               889
Name: count, dtype: int64
```

In [ ]:
```
# Define exchange rates for the 10 most popular currencies to EUR (accord
exchange_rates = {
    'EUR European Euro': 1.0,  # Base currency (no conversion needed)
    'USD\tUnited States dollar': 0.94,
    'GBP\tPound sterling  ': 1.15,
    'INR\tIndian rupee': 0.011,
    'CAD\tCanadian dollar': 0.7,
    'PLN\tPolish zloty': 0.22,
    'AUD\tAustralian dollar': 0.61,
    'BRL\tBrazilian real': 0.19,
    'SEK\tSwedish krona': 0.084,
    'CHF\tSwiss franc': 1.03
}

# Create a new column for converted salaries
df['Converted_Salary'] = 0.0

# Iterate through the DataFrame rows and apply the exchange rates
for index, row in df.iterrows():
    currency = row['Currency']
    salary = row['CompTotal']
    if currency in exchange_rates:
        conversion_rate = exchange_rates[currency]
        converted_salary = salary * conversion_rate
        df.at[index, 'Converted_Salary'] = converted_salary
```

In [ ]:
```
# Sample 15 random entries that were affected by the exchange rate and sh
converted_currency = df.loc[(df["Converted_Salary"] != 0) &
                            (df["Converted_Salary"].notna())]

display(converted_currency[["Currency", "CompTotal", "Converted_Salary"]]
```

	Currency	CompTotal	Converted_Salary
73403	EUR European Euro	45000.0	45000.0
47277	EUR European Euro	64000.0	64000.0
63919	EUR European Euro	65000.0	65000.0
58266	INR\tIndian rupee	2000000.0	22000.0
41508	PLN\tPolish zloty	600000.0	132000.0
76884	PLN\tPolish zloty	170000.0	37400.0
58785	USD\tUnited States dollar	159000.0	149460.0
30604	USD\tUnited States dollar	225000.0	211500.0
35828	USD\tUnited States dollar	135000.0	126900.0
15613	EUR European Euro	44000.0	44000.0
48659	USD\tUnited States dollar	30000.0	28200.0
47157	EUR European Euro	48000.0	48000.0
48028	EUR European Euro	60000.0	60000.0
3340	SEK\tSwedish krona	576000.0	48384.0
41119	USD\tUnited States dollar	100510.0	94479.4

In [ ]:
```
# As the highest entries within the "Information Services, IT, Software D

plt.figure(figsize=(12, 8))
plt.title('Boxplot of Converted Salaries by Industry')
sns.boxplot(x='Industry', y='Converted_Salary', data=converted_currency)
plt.yscale('log')  # Set y-axis to logarithmic scale
plt.xticks(rotation=-45, ha='left')
plt.show()
```

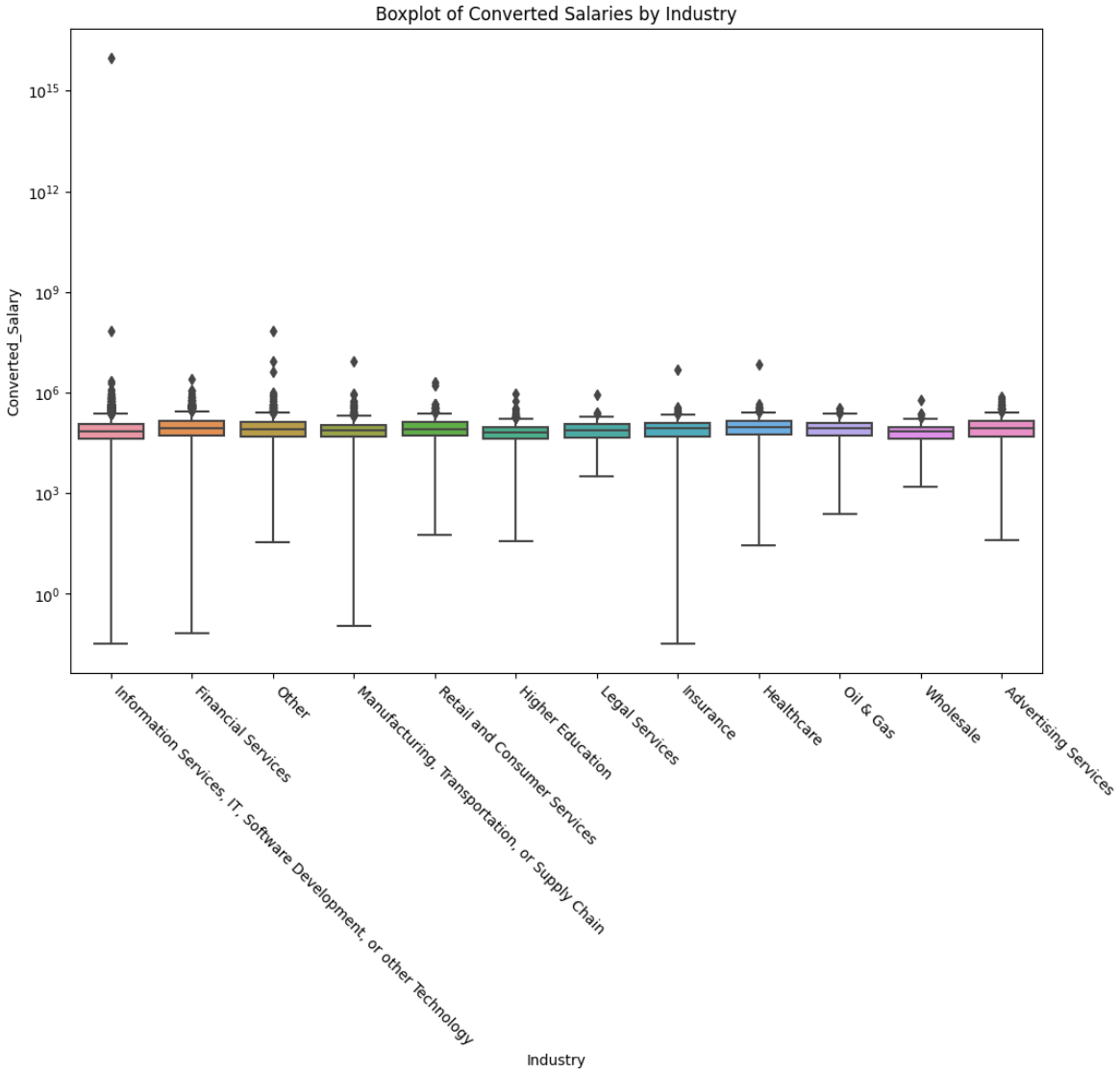

Figure 1: Boxplot of Converted Salaries by Industry Field

In [ ]:
```
# Group by 'Industry' and calculate the top 1% 'Converted_Salary'
max_comp_totals = converted_currency.groupby('Industry')['Converted_Salar

# Group by 'Industry' and calculate the median 'Converted_Salary'
median_comp_totals = converted_currency.groupby('Industry')['Converted_Sa

# Create a mapping of abbreviations for long industry names
abbreviations = {
    'Information Services, IT, Software Development, or other Technology'
    'Manufacturing, Transportation, or Supply Chain': 'Manufacturing, Log
}

# Replace long industry names with abbreviations
max_comp_totals['Industry'] = max_comp_totals['Industry'].replace(abbrevi
median_comp_totals['Industry'] = median_comp_totals['Industry'].replace(a

# Set the figure size
plt.figure(figsize=(12, 6))

# Calculate the number of industries
num_industries = len(max_comp_totals['Industry'])

# Define the width of each bar
```

```
bar_width = 0.35

# Define the x-axis positions for the bars
index = np.arange(num_industries)

# Create the grouped bar plots for median and top 1% values
plt.bar(index, median_comp_totals['Converted_Salary'], bar_width, label='
plt.bar(index + bar_width, max_comp_totals['Converted_Salary'], bar_width

# Set the x-axis labels, ticks, and rotation
plt.xlabel('Industry')
plt.ylabel('Total compensation [€]')
plt.title('Median and Top 1% Compensation by Industry')
plt.xticks(index + bar_width / 2, max_comp_totals['Industry'], rotation=4

# Add a legend
plt.legend()

# Define a threshold for the top 10%
threshold_max = max_comp_totals['Converted_Salary'].quantile(0.90)
threshold_median = median_comp_totals['Converted_Salary'].quantile(0.90)

# Apply conditional coloring to bars based on the threshold for both data
for i in range(num_industries):
    if max_comp_totals['Converted_Salary'][i] >= threshold_max:
        plt.gca().get_children()[i + num_industries].set_color('green')
    if median_comp_totals['Converted_Salary'][i] >= threshold_median:
        plt.gca().get_children()[i].set_color('green')  # Set color for m

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

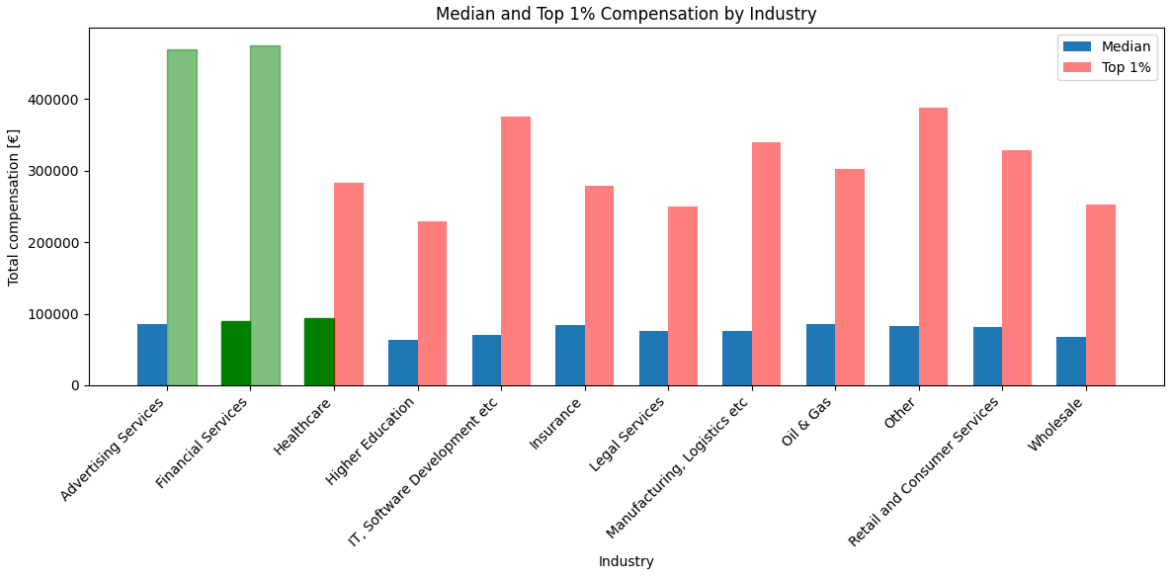

Figure 2: Median and Top 1% Compensation by Industry Field

In 2023 the most profitable industry field as a developer was Financial Services with the median and peak compensation being the highest among the listed fields. Closely followed Advertising Services with a comparably high top end compensation of 470k.

Median compensation showed less drastic differences, with Financial Services and Healthcare being in the top 10% of industry field.

It looks like the sector most closely connected to financial flows, also reaps the most fiscal gains in 2023.

Which type of Developer gets the highest compensation in 2023?

In [ ]:
```
# Group by 'DevType' and calculate the maximum 'Converted_Salary'
max_comp_totals = converted_currency.groupby('DevType')['Converted_Salary

# Group by 'Industry' and calculate the median 'Converted_Salary'
median_comp_totals = converted_currency.groupby('DevType')['Converted_Sal

# Create a mapping of abbreviations for long industry names
abbreviations = {
    'Information Services, IT, Software Development, or other Technology'
    'Manufacturing, Transportation, or Supply Chain': 'Manufacturing, Log
}

# Replace long industry names with abbreviations
max_comp_totals['DevType'] = max_comp_totals['DevType'].replace(abbreviat
median_comp_totals['DevType'] = median_comp_totals['DevType'].replace(abb

# Set the figure size
plt.figure(figsize=(12, 6))

# Calculate the number of Developer types
num_devs = len(max_comp_totals['DevType'])

# Define the width of each bar
bar_width = 0.35

# Define the x-axis positions for the bars
index = np.arange(num_devs)

# Create the grouped bar plots for median and top 1% values
plt.bar(index, median_comp_totals['Converted_Salary'], bar_width, label='
plt.bar(index + bar_width, max_comp_totals['Converted_Salary'], bar_width

# Set the x-axis labels, ticks, and rotation
plt.xlabel('DevType')
plt.ylabel('Total compensation [€]')
plt.title('Median and Top 1% Compensation by DevType')
plt.xticks(index + bar_width / 2, max_comp_totals['DevType'], rotation=45

# Add a legend
plt.legend()

# Define a threshold for the top 10%
threshold_max = max_comp_totals['Converted_Salary'].quantile(0.90)
threshold_median = median_comp_totals['Converted_Salary'].quantile(0.90)
```

```
# Apply conditional coloring to bars based on the threshold for both data
for i in range(num_devs):
    if max_comp_totals['Converted_Salary'][i] >= threshold_max:
        plt.gca().get_children()[i + num_devs].set_color('green')  # Set
    if median_comp_totals['Converted_Salary'][i] >= threshold_median:
        plt.gca().get_children()[i].set_color('green')  # Set color for m

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

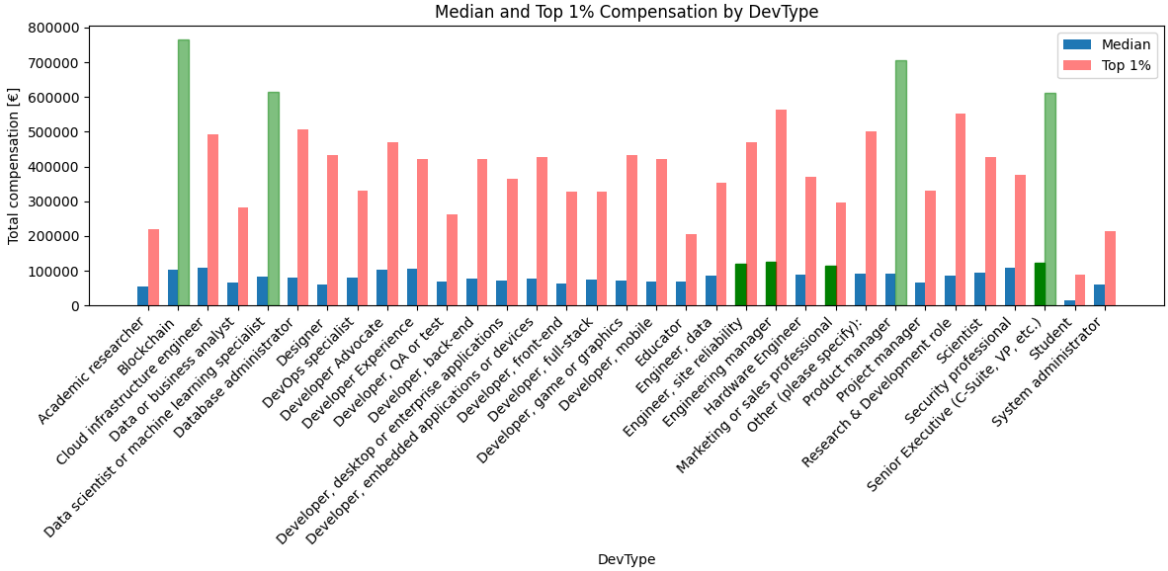

Figure 3: Median and Top 1% Compensation by Developer Type

How much does remote working matter to employees?

The covid pandemic forced a lot of companies into remote working to prevent infection of the work force and due to lockdown. Since the passing of the pandemic, life has gone more or less back to normal, but it has also shown that not all work activities have to be performed at the office. This begs the question:

Whats the distribution of remote work and education level?

Since the preference for remote, hybrid or in-person is not necessarily part of the survey, we have to take a roundabout way to answer this question.

Working Preference by Education Level

In [ ]:
```
# Define a custom function to calculate relative preferences within each
def calculate_relative_preferences(group, column_name):
```

```
    total = group[column_name].count()  # Total count of preferences in t
    relative_preferences = group['RemoteWork'].value_counts() / total  #
    return relative_preferences

def plot_work_preference_by_education(df, ax):
    # Create a mapping of abbreviations for long EdLevel names
    abbreviations = {
        'Associate degree (A.A., A.S., etc.)': 'Associate degree',
        "Bachelor's degree (B.A., B.S., B.Eng., etc.)": "Bachelor's degre
        "Master's degree (M.A., M.S., M.Eng., MBA, etc.)": "Master's degr
        "Primary/elementary school": "Primary/elementary school",
        "Professional degree (JD, MD, Ph.D, Ed.D, etc.)": "Professional d
        "Secondary school (e.g. American high school, German Realschule o
        "Some college/university study without earning a degree": "colleg
        "Something else": "Something else"
    }

    # Map the 'EdLevel' column to the abbreviated names
    df['Abbreviated_EdLevel'] = df['EdLevel'].map(abbreviations)

    # Group the data by 'EdLevel' and apply the custom function to calcul
    grouped_data = df.groupby('Abbreviated_EdLevel').apply(calculate_rela

    # Normalize each row so that the total height is 1.0
    normalized_data = grouped_data.div(grouped_data.sum(axis=1), axis=0)

    # Sort the DataFrame by the percentage of "Remote" in descending orde
    sorted_data = normalized_data.sort_values(by='Remote', ascending=Fals

    # Create a stacked bar chart with the sorted data
    sorted_data.plot(kind='bar', stacked=True, ax=ax)
    ax.set_xlabel('')
    ax.set_ylabel('Relative Preference')
    ax.set_title('Work Preference by Education Level', fontsize='large')
    ax.set_xticklabels(ax.get_xticklabels(), rotation=-45, ha='left')

# # Create a single figure and axis outside the function
# fig, ax = plt.subplots(figsize=(10, 6))

# # Call the function with the provided axis
# plot_work_preference_by_education(df, ax)

# # Place the legend outside the bar area
# ax.legend(title='Preference', bbox_to_anchor=(1.05, 1), loc='upper left

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

Working Preference by Industry Field

In [ ]:

```
def plot_work_preference_by_industry(df, ax):
    # Group the data by 'Industry' and apply the custom function to calcu
    grouped_data = df.groupby('Industry').apply(calculate_relative_prefer

    # Normalize each row so that the total height is 1.0
    normalized_data = grouped_data.div(grouped_data.sum(axis=1), axis=0)

    # Sort the DataFrame by the percentage of "Remote" in descending orde
    sorted_data = normalized_data.sort_values(by='Remote', ascending=Fals
```

```
    # Create a stacked bar chart with the sorted data
    sorted_data.plot(kind='bar', stacked=True, ax=ax)
    ax.set_xlabel('')
    ax.set_ylabel('Relative Preference')
    ax.set_title('Work Preference by Industry Field', fontsize='large')
    ax.set_xticklabels(ax.get_xticklabels(), rotation=-45, ha='left')

# # Create a single figure and axis outside the function
# fig, ax = plt.subplots(figsize=(10, 6))

# # Call the function with the provided axis
# plot_work_preference_by_industry(df, ax)

# # Place the legend outside the bar area
# ax.legend(title='Preference', bbox_to_anchor=(1.05, 1), loc='upper left

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

Working Preference by Developer Type

In [ ]:
```
def plot_work_preference_by_devtype(df, ax):
    # Group the data by 'DevType' and apply the custom function to calcul
    grouped_data = df.groupby('DevType').apply(calculate_relative_prefere

    # Calculate the percentage of people choosing 'Remote'
    percentage_remote = grouped_data['Remote'].mul(100)

    # Sort the dataframe based on the percentage of people choosing 'Remo
    sorted_index = percentage_remote.sort_values(ascending=False).index
    grouped_data_sorted = grouped_data.reindex(sorted_index)

    # Normalize each row so that the total height is 1.0
    normalized_data = grouped_data_sorted.div(grouped_data_sorted.sum(axi

    # Create a stacked bar chart
    normalized_data.plot(kind='bar', stacked=True, ax =ax)
    ax.set_xlabel('')
    ax.set_ylabel('Relative Preference')
    ax.set_title('Work Preference by Developer Type', fontsize='large')
    ax.set_xticklabels(ax.get_xticklabels(), rotation=-45, ha='left')

# # Create a single figure and axis outside the function
# fig, ax = plt.subplots(figsize=(20, 6))

# # Call the function with the provided axis
# plot_work_preference_by_devtype(df, ax)

# # Place the legend outside the bar area
# ax.legend(title='Preference', bbox_to_anchor=(1.05, 1), loc='upper left

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

Working Preference by Age

In [ ]:
```
def plot_work_preference_by_age(df, ax):
    # Group the data by 'Age' and apply the custom function to calculate
    grouped_data = df.groupby('Age').apply(calculate_relative_preferences

    # Calculate the percentage of people choosing 'Remote'
    percentage_remote = grouped_data['Remote'].mul(100)

    # Sort the dataframe based on the percentage of people choosing 'Remo
    sorted_index = percentage_remote.sort_values(ascending=False).index
    grouped_data_sorted = grouped_data.reindex(sorted_index)

    # Normalize each row so that the total height is 1.0
    normalized_data = grouped_data_sorted.div(grouped_data_sorted.sum(axi

    # Create a stacked bar chart
    normalized_data.plot(kind='bar', stacked=True, ax = ax)
    ax.set_xlabel('')
    ax.set_ylabel('Relative Preference')
    ax.set_title('Work Preference by Age', fontsize='large')
    ax.set_xticklabels(ax.get_xticklabels(), rotation=-45, ha='left')

# # Create a single figure and axis outside the function
# fig, ax = plt.subplots(figsize=(10, 6))

# # Call the function with the provided axis
# plot_work_preference_by_age(df, ax)

# # Place the legend outside the bar area
# ax.legend(title='Preference', bbox_to_anchor=(1.05, 1), loc='upper left

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

Working Preference by Organization Size

In [ ]:
```
def plot_work_preference_by_orgsize(df, ax):
    # Group the data by 'DevType' and apply the custom function to calcul
    grouped_data = df.groupby('OrgSize').apply(calculate_relative_prefere

    # Calculate the percentage of people choosing 'Remote'
    percentage_remote = grouped_data['Remote'].mul(100)

    # Sort the dataframe based on the percentage of people choosing 'Remo
    sorted_index = percentage_remote.sort_values(ascending=False).index
    grouped_data_sorted = grouped_data.reindex(sorted_index)

    # Normalize each row so that the total height is 1.0
    normalized_data = grouped_data_sorted.div(grouped_data_sorted.sum(axi

    # Create a stacked bar chart
    normalized_data.plot(kind='bar', stacked=True, ax =ax)
    ax.set_xlabel('')
    ax.set_ylabel('Relative Preference')
    ax.set_title('Work Preference by Organization Size', fontsize='large'
    ax.set_xticklabels(ax.get_xticklabels(), rotation=-45, ha='left')


# Create a single figure and axis outside the function
```

```
# fig, ax = plt.subplots(figsize=(10, 6))

# # Call the function with the provided axis
# plot_work_preference_by_orgsize(df, ax)

# # Place the legend outside the bar area
# ax.legend(title='Preference', bbox_to_anchor=(1.05, 1), loc='upper left

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

In [ ]:

```
fig3 = plt.figure(figsize=(30, 30))
gs = fig3.add_gridspec(3, 3, hspace=0.6)

f3_ax1 = fig3.add_subplot(gs[0, :])

f3_ax2 = fig3.add_subplot(gs[1, :2])

f3_ax3 = fig3.add_subplot(gs[1, 2])

f3_ax4 = fig3.add_subplot(gs[2, :2])

f3_ax5 = fig3.add_subplot(gs[2, 2])

# Plotting functions
plot_work_preference_by_devtype(df, f3_ax1)
plot_work_preference_by_industry(df, f3_ax5)
plot_work_preference_by_orgsize(df, f3_ax3)
plot_work_preference_by_age(df, f3_ax4)
plot_work_preference_by_education(df, f3_ax2)

# Disable legend for each subplot
f3_ax1.legend().set_visible(False)
f3_ax2.legend().set_visible(False)
f3_ax3.legend().set_visible(False)
f3_ax4.legend().set_visible(False)
f3_ax5.legend().set_visible(False)

# Create a common legend outside the subplots
legend_labels = ["In-Person", "Hybrid", "Remote"]  # Replace with your ac

# Adjust bbox_to_anchor to control the position of the legend more precis
fig3.legend(labels=legend_labels, loc='upper right', bbox_to_anchor=(0.15

plt.show()
```

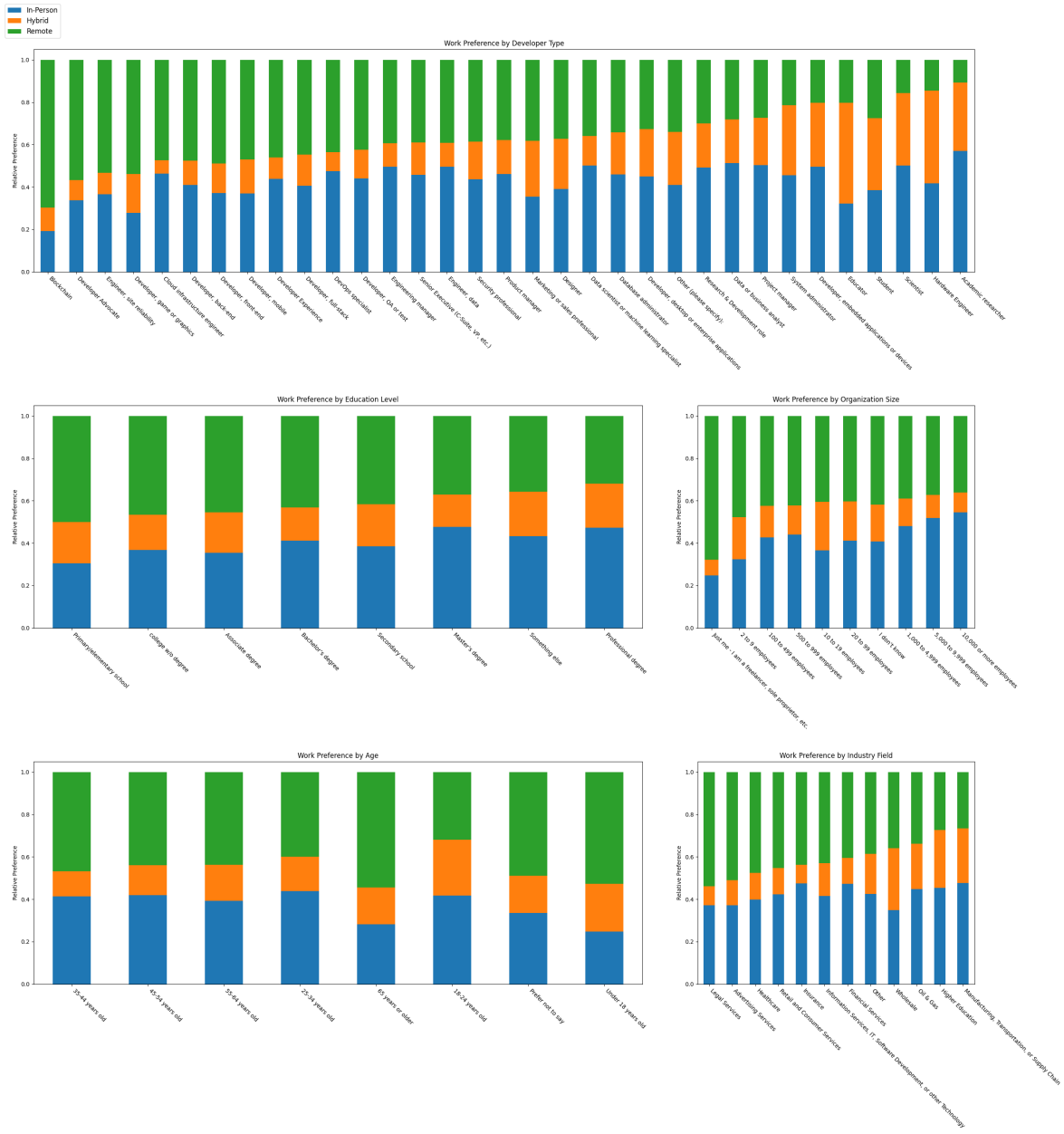

Figure 4: Work Preference by Developer Type, Education Level, Organization Size, Age & Industry Field

The figure above shows some of the metrics I thought might influence the proportion of people working remotely.
As the preference per-se is not mentioned in the survey, we can still make assumptions about peoples desire and possibility to work remotely.
The data above paints a picture in which people of newer Development branches, namely Blockchain, Cloud Infrastructure development and Developer Advocate work more remotely than established professions that are not unique to coding.
By this I mean the fields of teaching, education, hardware engineering and science, which have the among the lowest scores of remote work in the survey. This probably points towards a necessity of interacting with people or objects that can not be replaced by remote work.
Among educational Levels, the higher the education, the less people work remotely, this is probably a conjunction between the level of education and the type of developer these people represent. As jobs in hardware engineering science and

education rely on a certain academic degree, these data go hand in hand.
Its interesting to see that in smaller sized companies more people work remotely than in larger companies. On the lowest end, being a freelancer often doesn't come with the "luxury" of having an office space, which makes sense. On the other hand larger companies may be more reliant on people working in person to "ensure" a prosperous working environment by having employees coming into office.
Among the Age group people under the age of 18 and people over the age of 65 show a high frequency of remote work, probably due to independent or part-time employment, as these age groups are outside the common working age. Within the ages of 18 and 65 there is not a clear trend of remote or in-person work.
Among the different fields of Industry Legal Serives, Advertising and Healthcare show the highest frequency of remote work, maybe due to the contractory nature of these professions and a higher abundance of freelance work in these fields.
Professions in fields of Manufacturing, Transportation, Supply Chain, Higher Education and comparable jobs that rely on hands on contact with wares or people show the lowest frequency of remote work.

The survey allows for multiple entries in the Employment field. For better visualization I count the individual categories listed and create plots how the relate to work preferences.

How does coding experience affect the level of pay?

One would argue that experience in a field corresponds to higher salaries. Is this true for the profession of a developer in 2023?

In [ ]:
```
# Drop Entries without a Salary Information
salary_data = df['Converted_Salary'].dropna()

# Exclude zero values and limit the range of values to an interquartile r
salary_data = salary_data[(salary_data > salary_data.quantile(.05)) & (~s

# Convert "YearsCode" and "YearsCodePro" to numeric values
df['YearsCode'] = pd.to_numeric(df['YearsCode'], errors='coerce')
df['YearsCodePro'] = pd.to_numeric(df['YearsCodePro'], errors='coerce')

# Group by "YearsCode" and calculate median compensation
median_compensation = df.groupby('YearsCode')['CompTotal'].median().reset

# Group by "YearsCodePro" and calculate median compensation
```

```
median_compensation_pro = df.groupby('YearsCodePro')['CompTotal'].median(

# Create a figure
fig = plt.figure( figsize=(20, 12))
gs = fig.add_gridspec(2,2)

# Plot the first line on the first subplot
ax1 = fig.add_subplot(gs[0,:])
# Plot non professional experience in blue
ax1.plot(median_compensation['YearsCode'], median_compensation['CompTotal
# Plot professional experience in green
ax1.plot(median_compensation_pro['YearsCodePro'], median_compensation_pro
ax1.set_title('Salary by years of coding experience')
ax1.set_xlabel('Years of Coding Experience')
ax1.set_ylabel('Median Annual Compensation')
ax1.grid(axis='y', linestyle='--', alpha=0.7)
ax1.set_xticks(median_compensation['YearsCode'])
ax1.legend(loc='upper left')


# Plot histograms on the second subplot
ax2 = fig.add_subplot(gs[1,0])
ax2.hist([df['YearsCode'], df['YearsCodePro']], bins=10, color=['blue', '
ax2.set_title('Distribution of Coding Experience')
ax2.set_xlabel('Years of Coding Experience')
ax2.set_ylabel('Frequency')
ax2.grid(axis='y', linestyle='--', alpha=0.7)
ax2.legend()

# Plot histogram for the distribution of compensations
# Calculate bin width
bin_width = salary_data.max() / 20

# Generate integer bin edges
bin_edges = np.arange(salary_data.min(), salary_data.max(), bin_width).as

ax3 = fig.add_subplot(gs[1,1])
ax3.hist(salary_data, bins = bin_edges, color='green', edgecolor='black',
ax3.set_title('Histogram of Converted Salary (log scale)')
ax3.set_xlabel('Converted Salary')
ax3.set_ylabel('Frequency (log scale)')
ax3.grid(axis='y', linestyle='--', alpha=0.7)

# Add a common title for both subplots
plt.suptitle('Median Salary and Distribution by Years of Coding Experienc

plt.show()
```

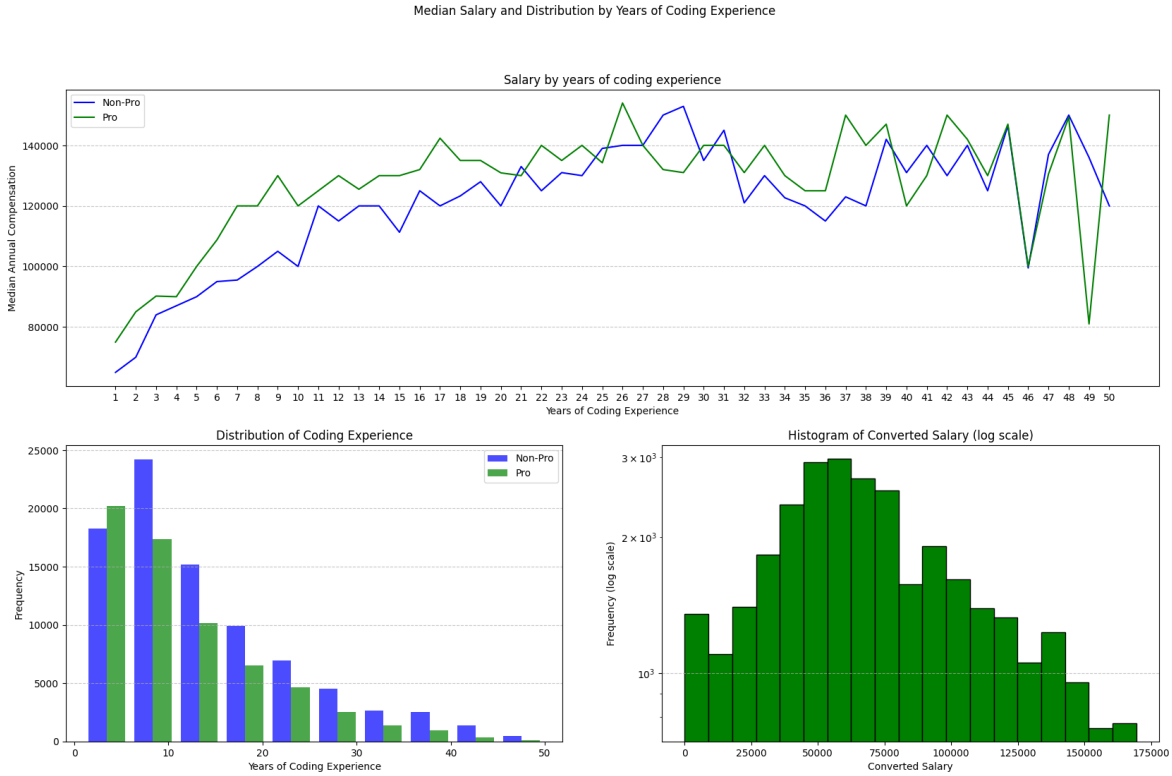

Figure 5: Median Salary and Distribution by Years of Coding Experience

As expected Junior Developers have the lowest compensation, but the first 10 years of coding experience rank up the salary by a good margin. Between 10 and 15 years of experience the salary don't change drastically and beyond 15 years of experience there is no longer a linear relationship between salary and experience. The histograms show the frequency of the respective answer options and, as can be seen, there are relatively few numbers for a period of experience of over 20 years, so that this area is only of limited significance.

Whats the most popular method to learn coding in 2023?

As I am a non-specialist programmer/data analyst myself, it is interesting for me personally to see how others learned to program in 2023.

In [ ]:
```
# Create binary columns for each category
categories = ['Books / Physical media',
              'Coding Bootcamp',
              'Colleague',
              'Friend or family member',
              'Hackathons (virtual or in-person)',
              'Online Courses or Certification',
              'On the job training',
              'Other online resources (e.g., videos, blogs, forum)',
              'School (i.e., University, College, etc)'
              ]

for category in categories:
```

```
    df[category] = df['LearnCode'].str.contains(category, case=False, na=

# Sum the counts for each category
grouped_data = df[categories].sum().reset_index()

# Sort the DataFrame by the count column in ascending order
grouped_data_sorted = grouped_data.sort_values(by=0, ascending=False)

# Create a bar plot
plt.figure(figsize=(10, 6))
plt.bar(grouped_data_sorted['index'], grouped_data_sorted[0], color='skyb
plt.ylabel('Count')
plt.title('Learning Sources Count')
plt.xticks(rotation=-45, ha='left')  # Rotate x-axis labels for better re

# Display the count values on top of the bars
for i, count in enumerate(list(grouped_data_sorted[0])):
    plt.text(i, count + 500, str(count), ha='center', va='bottom')

plt.show()
```

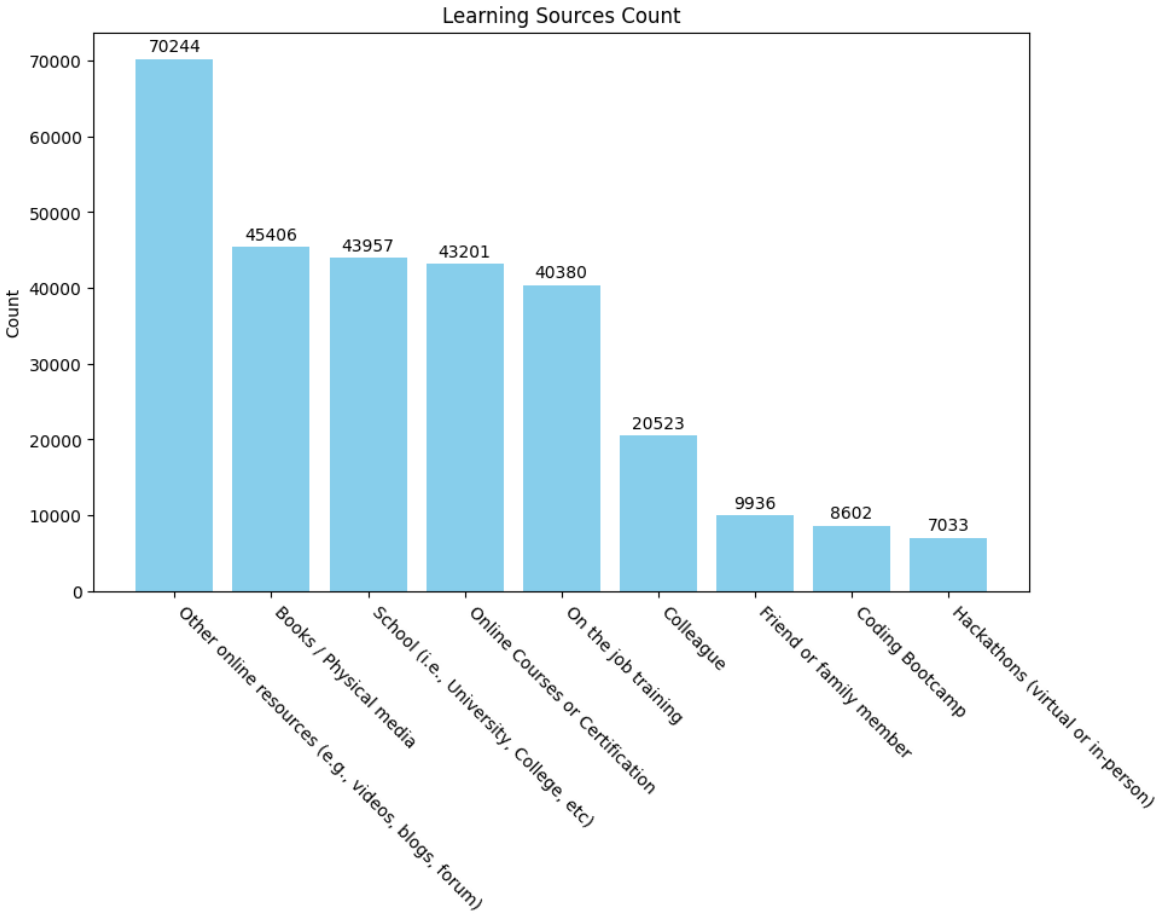

Figure 6: Coding Learning Sources 2023

Videos, blogs and forums were the most popular sources to learn coding in 2023, an asynchronous and mostly free way to learn coding in your spare time. Other more sophisticated like books, classes and online courses also made up a good portion. Coding Bootcamps and Hackathons were least popular, probably because they pose a decent barrier of entry, in that you need to be familiar with a language/ coding in general and that it is not easily compatible with a work/ school schedule.

In [ ]:
```
# Create binary columns for each category
categories = ['Technical documentation',
              'Blogs',
              'Online books',
              'Coding sessions (live or recorded)',
              'How-to videos',
              'Certification videos',
              'Video-based online courses',
              'Written-based online courses',
              'Auditory material (e.g., podcasts)',
              'Games that teach programming',
              'Online challenges (e.g., daily or weekly coding challenges
              'Interactive tutorial',
              'Written tutorials',
              'Online forum',
              'Stack Overflow'
              ]

for category in categories:
    df[category] = df['LearnCodeOnline'].str.contains(category, case=Fals

# Sum the counts for each category
grouped_data = df[categories].sum().reset_index()

# Sort the DataFrame by the count column in ascending order
grouped_data_sorted = grouped_data.sort_values(by=0, ascending=False)

# Create a bar plot
plt.figure(figsize=(10, 6))
plt.bar(grouped_data_sorted['index'], grouped_data_sorted[0], color='skyb
plt.ylabel('Count')
plt.title('Online Learning Sources Count')
plt.xticks(rotation=-45, ha='left')  # Rotate x-axis labels for better re

# Display the count values on top of the bars
for i, count in enumerate(list(grouped_data_sorted[0])):
    plt.text(i, count + 500, str(count), ha='center', va='bottom')

plt.show()
```

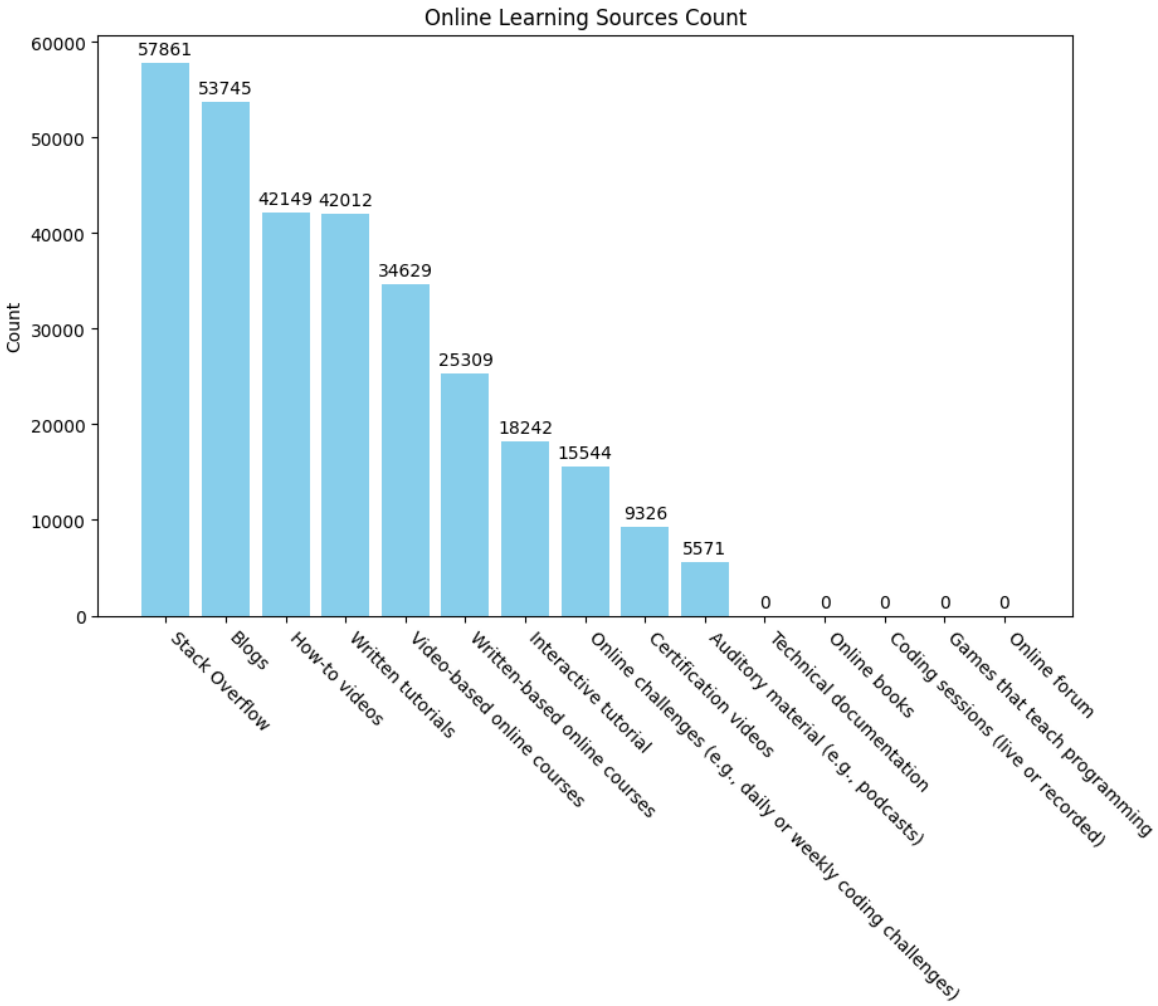

Figure 7: Online Coding Learning Sources 2023

Within the online learning sources, Stack Overflow, Blogs and how-to videos were the most frequently named sources. They offer mostly precise and correct answers to a specialized question instead of the more broad scope a book or a technical documentation might offer. 5 of the categories had a frequency of 0, meaning nobody selected these answers. It doesn't seem feasable that nobody picked these options, but they dont show up in the cleaned dataset.

Are you more likely to get a job as a developer if you have a master's degree?

Since I have a background in Neuroscience and finished my studies with a Master's Degree I was interested to see if having a degree increases the chance to get a job as a developer.
To answer this question I split the column of Education level into 2 groups ("Master or higher" & "Bachelor or lower") and displayed the frequency of each Type of Developer in each group.

In [ ]:
```
def categorize_education(ed_level):
    if "Master's degree (M.A., M.S., M.Eng., MBA, etc.)" in ed_level or '
        return 'Master or Higher'
    else:
        return 'Bachelor or Lower'

# Replace NaN values in 'EdLevel' with 'Bachelor or Lower'
df['EdLevel'].fillna('Bachelor or Lower', inplace=True)

# Apply the function to create a new column 'EdLevelCategory'
df['EdLevelCategory'] = df['EdLevel'].apply(categorize_education)

# Group by 'EdLevelCategory' and 'DevType', then count the occurrences fo
grouped_data_all = df.groupby(['EdLevelCategory', 'DevType']).size().rese

# Create subplots
fig, axs = plt.subplots(2, 1, figsize=(12, 10))

# Subplot 1: Plot all DevTypes using seaborn
sns.barplot(ax=axs[0], x='EdLevelCategory', y='Count', hue='DevType', dat
axs[0].set_title('Distribution of Developer Types by Education Level Cate
axs[0].set_yscale('log')  # Set y-axis to logarithmic scale
axs[0].set_xlabel("")

# Place the legend outside the bar area
axs[0].legend(title='DevType', bbox_to_anchor=(-0.1, 1), loc='upper right

# Subplot 2: Plot DevTypes where more people have a Masters Degree
master_counts = grouped_data_all[grouped_data_all['EdLevelCategory'] == '
bachelor_counts = grouped_data_all[grouped_data_all['EdLevelCategory'] ==

selected_devtypes = master_counts[master_counts > bachelor_counts].index.
selected_data = df[(df['EdLevelCategory'].isin(['Master or Higher', 'Bach
grouped_data_subset = selected_data.groupby(['EdLevelCategory', 'DevType'

# Get handles and labels from subplot 1 legend
handles, labels = axs[0].get_legend().legend_handles, [t.get_text() for t

# Create a dictionary mapping DevTypes to colorsProgramming languages and
color_palette_all = {label: handle.get_facecolor() for label, handle in z

# Plot using seaborn
sns.barplot(ax=axs[1], x='EdLevelCategory', y='Count', hue='DevType', dat

# Get the handles and labels from subplot 2 legend
handles_subset, labels_subset = axs[1].get_legend().legend_handles, [t.ge

# Create a dictionary mapping DevTypes to colors for subplot 2
color_palette_subset = {label: color_palette_all.get(label, (0.8, 0.8, 0.

# Subplot 2: Plot DevTypes where more people have a Masters Degree
ax2 = sns.barplot(ax=axs[1], x='EdLevelCategory', y='Count', hue='DevType

# Remove the Seaborn legend
ax2.get_legend().remove()

# Create a custom legend using matplotlib
legend_labels = []
for devtype, color in color_palette_subset.items():
```

```
    patch = plt.Line2D([0], [0], marker='o', color='w', label=devtype, ma
    legend_labels.append(patch)

# Add the legend to subplot 2
axs[1].legend(handles=legend_labels, title='DevType', bbox_to_anchor=(1.0
axs[1].set_xlabel("")
axs[1].set_title('Distribution of Developer Types by Education Level Cate

# Increase space between subplots
fig.subplots_adjust(hspace=0.25)

plt.show()
```

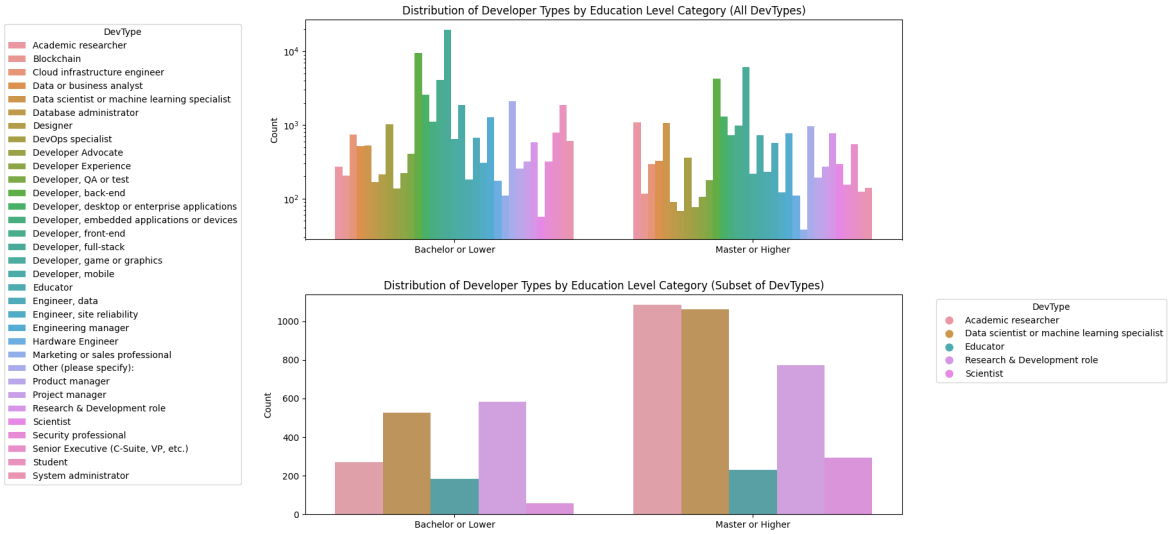

Figure 8: Distribution of Developer Types by Education Level Category

The only Type of Developers that show a higher frequency of Master or Higher Education are those from academia, where a certain degree is necessary.

Which are the most popular languages of 2023?

Programming languages and frameworks are being released ever more frequently, but do these new and shiny languages also find their way into actual applications people are working on?

Since the survey allows for multiple programming language entries per participant I firstly create binary columns that count every occurance of a language.

In [ ]:
```
# Set a more visually appealing color palette
sns.set_palette("viridis")

# Function to create binary columns
def create_binary_columns(df, categories, column_name):
    for category in categories:
        df.loc[:, category] = df[column_name].str.contains(category, case
```

```
# Function to plot bar subplot
def plot_bar_subplot(ax, data, title, ylim=None, value_counts_offset=500)
    ax.bar(data['index'], data[0], alpha=0.8)
    ax.set_ylabel('Count', fontsize=12)
    ax.set_title(title, fontsize=14)
    ax.set_xticks(range(len(data['index'])))
    ax.set_xticklabels(data['index'], rotation=45, ha='right', fontsize=1
    if ylim:
        ax.set_ylim(ylim[0], ylim[1])

    for i, count in enumerate(data[0]):
        offset = value_counts_offset if 'Data or Business Analyst' in tit
        ax.text(i, count + offset, str(count), ha='center', va='bottom',

# Create binary columns for each category
categories = ["Ada", "Apex", "APL", "Assembly", "Bash/Shell (all shells)"
              "Crystal", "Dart", "Delphi", "Elixir", "Erlang", "F#", "Flo
              "Haskell", "HTML/CSS", "Java", "Javascript", "Julia", "Kotl
              "Objective-C", "OCaml", "Perl", "PHP", "PowerShell", "Prolo
              "SAS", "Scala", "Solidity", "SQL", "Swift", "TypeScript", "

# Create subplots
fig, axs = plt.subplots(2, 2, figsize=(15, 8), gridspec_kw={'hspace': 0.8

# Plot the first subplot Languages have worked with
create_binary_columns(df, categories, 'LanguageHaveWorkedWith')
grouped_data_sorted = df[categories].sum().reset_index().sort_values(by=0
plot_bar_subplot(axs[0, 0], grouped_data_sorted, 'Top 10 Popular language

# Plot the second subplot Languages you want to work with next Year
create_binary_columns(df, categories, 'LanguageWantToWorkWith')
grouped_data_sorted = df[categories].sum().reset_index().sort_values(by=0
plot_bar_subplot(axs[0, 1], grouped_data_sorted, 'Prediction Top 10 Langu

# Prepare third plot Popular languages for Data or Business Analysts
# Filter data for "Data or business analyst"
filtered_df = df[df['DevType'] == "Data or business analyst"]
create_binary_columns(filtered_df, categories, 'LanguageHaveWorkedWith')
grouped_data_sorted = filtered_df[categories].sum().reset_index().sort_va
plot_bar_subplot(axs[1, 0], grouped_data_sorted, 'Top 10 Popular language

# Prepare fourth plot Popular languages for Data or Business Analysts in
create_binary_columns(filtered_df, categories, 'LanguageWantToWorkWith')
grouped_data_sorted = filtered_df[categories].sum().reset_index().sort_va
plot_bar_subplot(axs[1, 1], grouped_data_sorted, 'Top 10 Popular language

plt.show()
```

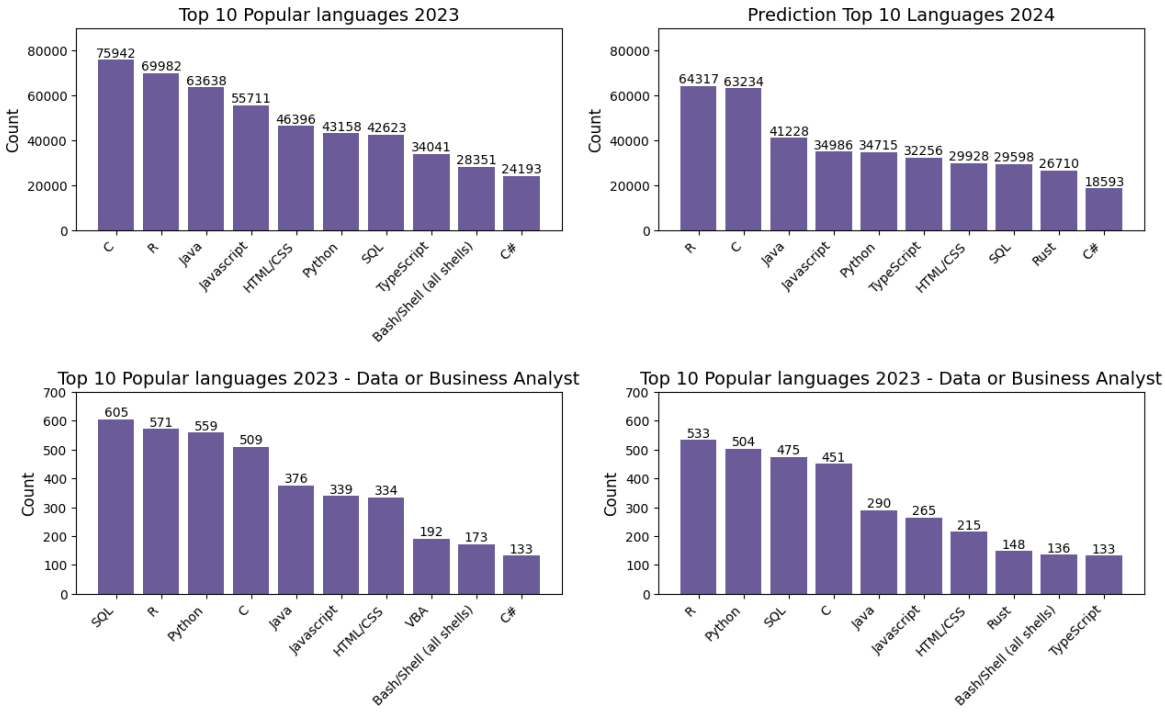

Figure 9: Popular languages in 2023 and predictions for 2024

Among the most popular languages we have the jack of all trades languages, that can be applied to most applications like C, Java, Python and the Web languages Javascript and HTML/ CSS, which is to be expected as these languages find application in most fields. Whats interesting for me is the prevalence of R, a language for statistical analysis, graphics representation and reporting, which is also the #1 language people want to work with in 2024.
If we focus our interest on the stack used by data and business analysts we see a similiar picture with R, Python, C and Java being in the top 5. Especially interesting for the field of data analysis is SQL, a language used for managing data held in a relational database management system.