Direct causes

Parents living in suburban believe their children spend more time on social media. We can see that families living in suburban have more children who are staying at home.

```
import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd
import numpy as np

# Load data
data = pd.read_csv('pre_processed_data.csv')

# Add some jitter (random noise) to the 'No_of_kids_at_home_0_11' values
to spread the points out for visibility
data['No_of_kids_at_home_0_11_jittered'] = data['No_of_kids_at_home_0_11']
+ np.random.normal(0, 0.1, size=len(data))

# Add some jitter to the 'No_of_kids_at_home_above_12' values as well
data['No_of_kids_at_home_above_12_jittered'] =
data['No_of_kids_at_home_above_12'] + np.random.normal(0, 0.1,
size=len(data))

# Define marker styles for different 'TS_on_socialmedia' categories
markers = {
    'Too much time': 'X',
    'About the right amount of time': 'o',
    'Too little time': 's',
    'Refused': 'd'
}

# Create a scatter plot
plt.figure(figsize=(10, 6))
sns.scatterplot(
    data=data,
    x='P_neighborhood',  # Neighborhoods will be on the x-axis
    y='No_of_kids_at_home_0_11_jittered',  # Use jittered values for y-
axis
    style='TS_on_socialmedia',  # Use different markers for social media
time categories
    hue='No_of_kids_at_home_above_12',  # Use a color gradient for the
number of older kids
    markers=markers,  # Set marker styles
    palette='coolwarm',  # Color palette for older kids
    s=100,  # Uniform size for all markers
    alpha=0.6  # Semi-transparency for markers
)

# Add a legend outside the plot
plt.legend(bbox_to_anchor=(1.05, 1), loc='upper left', borderaxespad=0.,
title='Time Spent on Social Media')
```

```
# Add titles and labels
plt.title('Social Media Usage and Number of Kids at Home by Neighborhood')
plt.xlabel('Neighborhood')
plt.ylabel('Number of Kids at Home Aged 0-11')

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

We can find that most parents who live in the suburban work full time.

```
'''
Parent job type
'''
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# Load data
data = pd.read_csv('pre_processed_data.csv')

# Create the picture
plt.figure(figsize=(10, 6))
sns.histplot(data, x='P_employment_type', hue='P_neighborhood',
multiple='stack')
plt.title('Distribution of Age own a phone by living area')
plt.xlabel('Parent job type')
plt.ylabel('Frequency')
plt.xticks(rotation=0, fontsize=12)  # Rotate the degree and change the
word size
plt.yticks(fontsize=12)
plt.tight_layout()
plt.show()
```

Percentage of Responses for Different Technologies Usage

```
import pandas as pd
import matplotlib.pyplot as plt

# Load the CSV file
file_path = 'pre_processed_data.csv'
data = pd.read_csv(file_path)
```

```
# Selecting and renaming the required columns
columns_of_interest = ['CHD_smartphone', 'CHD_voice_assistant',
'CHD_gaming_console', 'CHD_tablet', 'CHD_computer', 'CHD_tv']
display_column_names = {'CHD_smartphone': 'Smartphone',
'CHD_voice_assistant': 'Voice Assistant', 'CHD_gaming_console': 'Gaming
Console', 'CHD_tablet': 'Tablet', 'CHD_computer': 'Computer', 'CHD_tv':
'TV'}
data_selected =
data[columns_of_interest].rename(columns=display_column_names)

# Calculating the percentage of 'Yes' and 'No' for each column
percentages = {}
for col in display_column_names.values():
    counts = data_selected[col].value_counts(normalize=True)
    percentages[col] = {
        'Yes': counts.get('Yes, my child uses or interacts with this', 0)
* 100,
        'No': counts.get('No, my child does not use or interact with
this', 0) * 100
    }

# Creating a DataFrame for easy plotting
percentages_df = pd.DataFrame(percentages).T

# Plotting with percentage annotations
fig, ax = plt.subplots(figsize=(8, 4))  # Adjusted to a smaller size
percentages_df.plot(kind='bar', stacked=True, ax=ax)

# Adding the percentage annotations
for p in ax.patches:
    width = p.get_width()
    height = p.get_height()
    x, y = p.get_xy()
    ax.annotate(f'{height:.1f}%', (x + width/2, y + height/2),
ha='center')

# Setting the plot details
plt.title('Percentage of Responses for Different Technologies Usage',
fontsize=12)
plt.xlabel('Technologies', fontsize=10)
plt.ylabel('Percentage (%)', fontsize=10)
plt.xticks(rotation=45, fontsize=8)
plt.yticks(fontsize=8)
plt.legend(title='Response', fontsize=8)
plt.tight_layout()

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

Online Concerns: Parental Perspectives

```
import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd
import numpy as np


# Placeholder data for demonstration
data = pd.DataFrame({
    'CO_online_bully': ['Somewhat concerned', 'Very concerned', 'Not at
all concerned', 'Not too concerned', 'Somewhat concerned'],
    'CO_screen_time': ['Not too concerned', 'Somewhat concerned', 'Very
concerned', 'Not at all concerned', 'Somewhat concerned'],
    'CO_sexual_explicit_content': ['Very concerned', 'Somewhat concerned',
'Not at all concerned', 'Not too concerned', 'Very concerned'],
    'CO_online_fraud': ['Not at all concerned', 'Very concerned',
'Somewhat concerned', 'Not too concerned', 'Somewhat concerned'],
    'CO_violent_content': ['Somewhat concerned', 'Not too concerned',
'Very concerned', 'Not at all concerned', 'Somewhat concerned']
})

# Convert response categories to numeric
concern_mapping = {
    'Very concerned': 3,
    'Somewhat concerned': 2,
    'Not too concerned': 1,
    'Not at all concerned': 0
}

# Apply mapping and transpose the dataframe for the heatmap
heatmap_data = data.replace(concern_mapping).transpose()

# Compute the proportion of each response category for the heatmap
heatmap_data = heatmap_data.apply(lambda x:
x.value_counts(normalize=True), axis=1).fillna(0)

# Define a custom color map for a black to purple gradient
cmap = sns.color_palette("Purples", as_cmap=True)

# Create the heatmap
plt.figure(figsize=(12, 7))
sns.heatmap(heatmap_data, annot=True, cmap=cmap, fmt=".2f")

# Add titles and labels
plt.title('Proportion of Concern Levels for Various Online Concerns')
plt.xlabel('Concern Levels')
plt.ylabel('Online Concerns')

# Adjust the x-axis to show the response categories in the desired order
response_categories = ['Not at all concerned', 'Not too concerned',
'Somewhat concerned', 'Very concerned']
```

```
plt.xticks(ticks=np.arange(len(response_categories)) + 0.5,
labels=response_categories, rotation=0)

# Adjust the y-axis to remove 'CO_' prefix and display the concerns
concern_labels = [label.replace('CO_', '') for label in
heatmap_data.index]
plt.yticks(ticks=np.arange(len(concern_labels)) + 0.5,
labels=concern_labels, rotation=0)

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

Sources of Screen Time Advice for Parents

```
import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd

# Placeholder data for demonstration
data = pd.DataFrame({
    'advice_socialmedia': ['Yes', 'No', 'Yes', 'Yes', 'No'],
    'advice_online_blogs': ['No', 'Yes', 'Yes', 'No', 'No'],
    'advice_other_parents': ['Yes', 'Yes', 'No', 'Yes', 'Yes'],
    'advice_books_magazines': ['No', 'No', 'Yes', 'No', 'Yes'],
    'advice_doctors': ['Yes', 'Yes', 'No', 'Yes', 'No'],
    'advice_teachers': ['No', 'No', 'Yes', 'No', 'Yes']
})

# Convert 'Yes'/'No' answers to binary values 1/0
binary_mapping = {'Yes': 1, 'No': 0}
for column in data.columns:
    data[column] = data[column].map(binary_mapping)

# Calculate the proportion of 'Yes' responses for each advice source
proportion_yes = data.mean()

# Calculate the proportion of 'No' responses
proportion_no = 1 - proportion_yes

# Combine the two into a DataFrame
heatmap_data = pd.concat([proportion_yes, proportion_no], axis=1)
heatmap_data.columns = ['Yes', 'No']

# Define a custom black-purple-black color map
cmap = sns.color_palette(["black", "#4A148C", "black", "#4A148C"],
as_cmap=True)

# Create the heatmap
```

```
plt.figure(figsize=(10, 6))
sns.heatmap(heatmap_data, annot=True, cmap=cmap, fmt=".2f")

# Add titles and labels
plt.title('Proportion of Respondents Saying Yes/No to Getting Screen Time
Advice from Various Sources')
plt.xlabel('Response')
plt.ylabel('Sources of Advice')

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

Modified these two pieces of code

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
from matplotlib.ticker import PercentFormatter
from sklearn.preprocessing import LabelEncoder

# Load data
data = pd.read_csv('pre_processed_data.csv')

# Rename 'P_race' to 'Race' in the DataFrame
data.rename(columns={'P_race': 'Race'}, inplace=True)

# Continue with the visualization or other analysis
dfm = data[['ADV_socialmedia', 'ADV_online_blogs', 'ADV_other_parents',
'ADV_books_magazines', 'ADV_doctors',
'ADV_teachers']].melt(var_name='advice_from', value_name='response')

# Create a categorical plot with the new 'Race' column as the y-axis
sns.catplot(kind='swarm', data=data, x='Child_owns_smartphone_age',
y='Race', order=['0-2', '3-4', '5-8', '9-11'],
hue='Child_first_age_smartphone', aspect=1.6)

plt.show()
```

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
from matplotlib.ticker import PercentFormatter
from sklearn.preprocessing import LabelEncoder
```

```
data = pd.read_csv('pre_processed_data.csv')

if 'P_employment_type' in data.columns:
    data.rename(columns={'P_employment_type': 'employment_type'},
inplace=True)

dfm = data[['ADV_socialmedia', 'ADV_online_blogs', 'ADV_other_parents',
'ADV_books_magazines', 'ADV_doctors',
'ADV_teachers']].melt(var_name='advice_from', value_name='response')

sns.catplot(kind='swarm', data=data, x='Child_owns_smartphone_age',
y='employment_type', order=['0-2', '3-4', '5-8', '9-11'],
hue='Child_first_age_smartphone')
plt.show()
```