Data Preparation dataset Healthcare Workforce Mental Health

Disusun untuk memenuhi tugas 1 mata kuliah Pembelajaran Mesin

Oleh:

Willy Jonathan Arsyad (2208107010037)
Agil Mughni (2208107010025)
Alfi Zamriza (2208107010080)
T.M Fadlul Ihsan (2208107010088)
M. Arkan Haris (2208107010022)

JURUSAN INFORMATIKA
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS SYIAH KUALA
DARUSSALAM, BANDA ACEH
2025

1. Data Description

Mengenai dataset Healthcare Workforce Mental Health, yang diambil dari website kaggle, dataset tersebut terdiri dari 10 kolom dan 5.000 baris data dalam format file ‘csv’. Dimana dari ke-10 kolom dari dataset tersebut berupa;

1. Employee ID → ID unik masing-masing karyawan (terdapat 5.000 ID)
2. Employee Type → Peran Karyawan dalam sistem kesehatan (Terdapat 10 peran, seperti *Physician, Medical Assistant*)
3. Department → Departemen tempat bekerja (Terdapat 10 departemen, seperti *ICU, Pediatrics*)
4. Workplace Factor → Faktor utama yang mempengaruhi pekerja (Terdapat 8 faktor, seperti *Heavy Workload, Poor Work Environment*)
5. Stress Level → Tingkat stress dari karyawan (dalam skala 1 - 10)
6. Burnout Frequency → Tingkat frekuensi kelelahan karyawan secara mental ('Often', *'Occasionally', 'Never'*)
7. Job Satisfaction → Tingkat Kepuasan terhadap pekerjaan (dalam skala 1-5, dimana 1 berarti sangat tidak puas, dan 5 berarti sangat puas)
8. Access to EAPs → Apakah karyawan memiliki akses untuk program asisten karyawan (*'Yes', 'No'*)
9. Mental Health Absences → Total hari cuti yang diambil dengan masalah kesehatan mental
10. Turnover Intention → Apakah karyawan berkeinginan untuk berhenti? (*'Yes', 'No'*)

2. Data Loading:

Dataset diolah dengan menggunakan python pada environment Google Collab, sehingga pembacaan data dilakukan dengan membaca dataset dengan perantara github dan membaca file ‘csv’ dari kaggle yang telah diupload. Dataset dibaca menggunakan library ‘pandas’ dan disimpan dalam variabel untuk memudahkan tahapan preprocessing berikutnya.

3. Data Understanding:

Pada tahapan pemrosesan ini ada cukup banyak hal untuk dilakukan, namun kami mulai dengan proses pengecekan data yang telah dibaca sebelumnya. Dataset dicek kembali jumlah baris, dan kolom yang terbaca, yaitu 5.000 baris, dengan 10 kolom. Kemudian, kami menampilkan tipe data dari masing-masing kolom dataset yang telah dibaca, dan memastikan bahwa tipe data yang ditetapkan secara otomatis memang sudah tepat.

```
[4] raw_df.shape

(5000, 10)

[5] raw_df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5000 entries, 0 to 4999
Data columns (total 10 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   Employee ID           5000 non-null   object
 1   Employee Type         5000 non-null   object
 2   Department            5000 non-null   object
 3   Workplace Factor      5000 non-null   object
 4   Stress Level          5000 non-null   int64
 5   Burnout Frequency     5000 non-null   object
 6   Job Satisfaction      5000 non-null   int64
 7   Access to EAPs        5000 non-null   object
 8   Mental Health Absences 5000 non-null  int64
 9   Turnover Intention    5000 non-null   object
dtypes: int64(3), object(7)
memory usage: 390.8+ KB
```

Kemudian, kami melakukan analisis statistik terhadap dataset tersebut. Diantaranya yaitu analisis rata-rata, standar deviasi, kuartil, beserta nilai maksimum dan minimum dari masing-masing kolom pada dataset. Tak lupa juga kami tampilkan analisis terhadap kolom-kolom non-numerik yang tidak dapat dianalisis menggunakan beberapa metode sebelumnya. Ada pula, informasi yang dapat kami simpulkan dari analisis tambahan ini seperti banyaknya unique value dari setiap kolom dataset, dan juga label dengan jumlah kemunculan terbanyak, beserta jumlah kemunculannya.

```
[6] raw_df.describe(include="number")
```

	Stress Level	Job Satisfaction	Mental Health Absences
count	5000.000000	5000.000000	5000.000000
mean	7.327800	2.202200	7.396200
std	1.407673	1.045722	2.878625
min	4.000000	1.000000	0.000000
25%	7.000000	1.000000	5.000000
50%	8.000000	2.000000	7.000000
75%	8.000000	3.000000	9.000000
max	9.000000	5.000000	19.000000

```
[7] raw_df.describe(include="object")
```

	Employee ID	Employee Type	Department	Workplace Factor	Burnout Frequency	Access to EAPs	Turnover Intention
count	5000	5000	5000	5000	5000	5000	5000
unique	5000	10	10	8	3	2	2
top	HCP-00001	Registered Nurse	General Medicine	Heavy Workload	Often	Yes	Yes
freq	1	1283	1283	2138	2221	3594	3335

Selanjutnya, kami juga memeriksa jumlah null value, ataupun data dengan entry yang sama (duplikat). Namun setelah pemeriksaan, tidak ditemukan adanya baris dengan multiple entry ataupun memiliki null value, sehingga dataset ini tidak kami lakukan pengurangan atau pembuangan baris dari dataset, ataupun penyesuaian lainnya seperti pengisian nilai dari kolom berdasarkan nilai rata-rata, atau median.

```
[8] raw_df.isna().sum()
```

	0
Employee ID	0
Employee Type	0
Department	0
Workplace Factor	0
Stress Level	0
Burnout Frequency	0
Job Satisfaction	0
Access to EAPs	0
Mental Health Absences	0
Turnover Intention	0

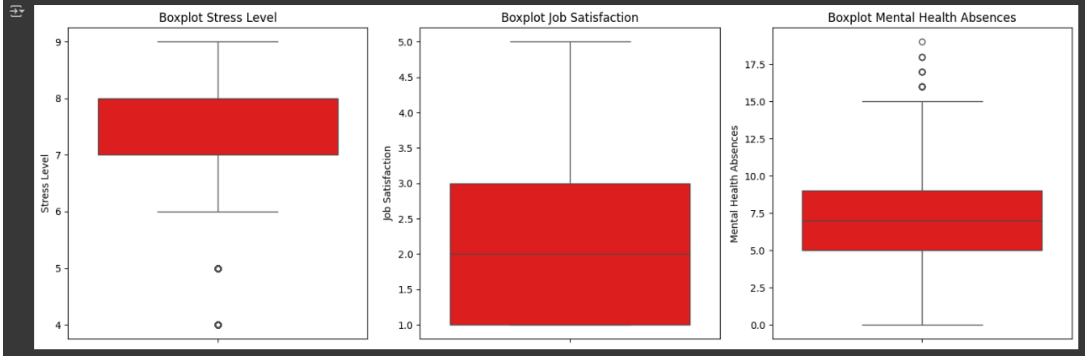
dtype: int64

```
raw_df.duplicated().sum()
```

```
0
```

Berdasarkan boxplot yang ditampilkan, tingkat stres karyawan cenderung berada pada level sedang hingga tinggi dengan median sekitar 7. Kepuasan kerja rata-rata tergolong rendah dengan median sekitar 2.0. Data ini menunjukkan adanya potensi hubungan antara tingkat stres, kepuasan kerja, dan kesehatan mental karyawan. Namun, analisis lebih lanjut diperlukan untuk mengkonfirmasi hubungan tersebut dan mengidentifikasi faktor-faktor lain yang berkontribusi.

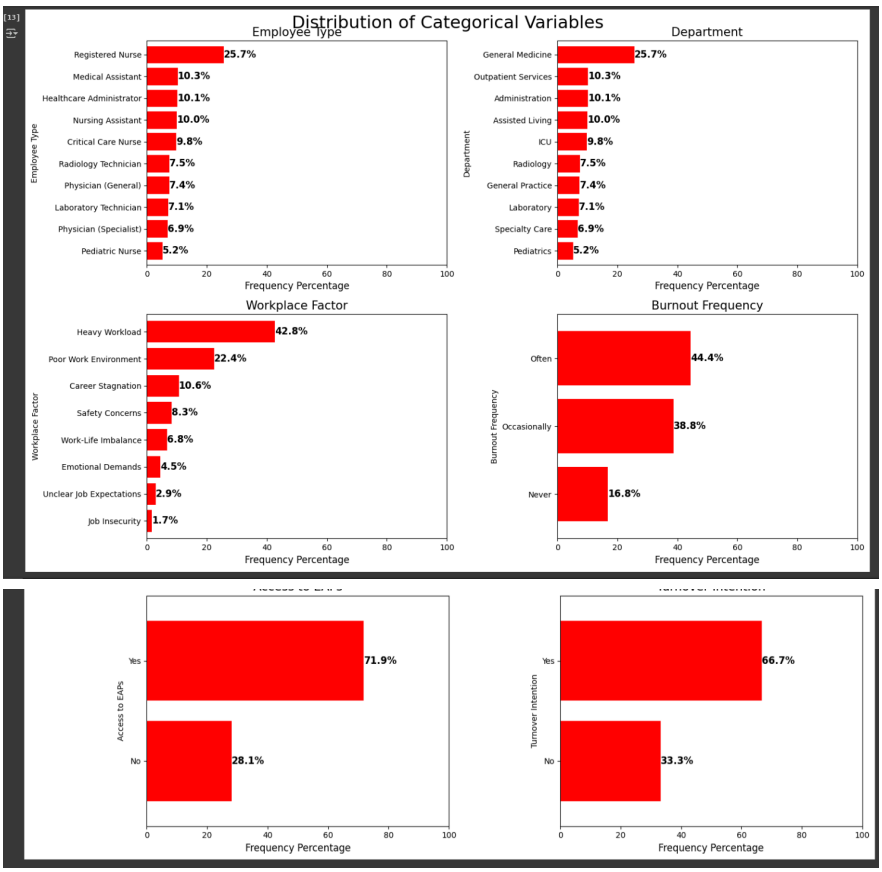
```
# Membuat subplot grid untuk setiap fitur numerik
numerical_columns = raw_df.select_dtypes(include=np.number).columns
num_cols = len(numerical_columns)
fig, axes = plt.subplots(nrows=(num_cols + 2) // 3, ncols=3, figsize=(15, 5 * ((num_cols + 2) // 3)))

# Loop untuk setiap fitur numerik
for i, col in enumerate(numerical_columns):
    sns.boxplot(y=raw_df[col], ax=axes.flatten()[i], color='red')
    axes.flatten()[i].set_title(f'Boxplot {col}')

plt.tight_layout()
plt.show()
```



Berdasarkan visualisasi yang kami tampilkan selanjutnya, terlihat bahwa dalam skala 1-10, persebaran tingkat stress dari dataset berada pada rentang 4 hingga 9. Dimana tingkat stress 8-8.5 memiliki jumlah karyawan terbanyak dengan jumlah 1513 karyawan, dengan kurva yang dihasilkan yaitu merupakan Left Skew yang terkonsentrasi pada bagian kanan kurva. Kemudian untuk Tingkat kepuasan pekerjaan 1.8-2.2 memiliki jumlah karyawan terbanyak yaitu pada 1834 karyawan dengan kurva dengan tipe Right Skew yang terkonsentrasi pada bagian kiri kurva yang menggunakan skala 1-5. Berbeda dengan kurva terhadap faktor total cuti yang diambil oleh karyawan yang memiliki distribusi relatif normal. Dengan jumlah sekitar 5.7-6.5 hari yang diambil oleh 1380 karyawan.

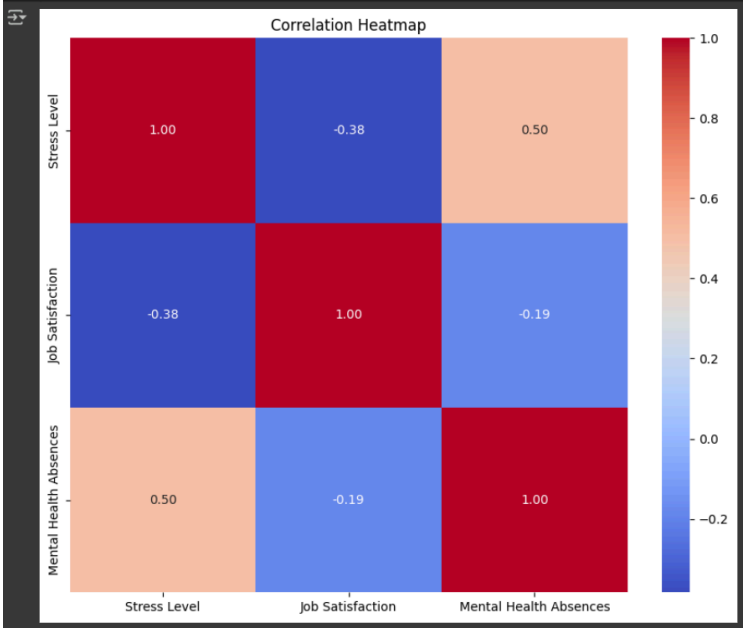
Kemudian, dapat dilihat bahwa sebagian besar data kategorik dari dataset memiliki perbedaan yang relatif cukup signifikan antara kategori yang memiliki jumlah terbanyak dan kedua terbanyak. Hal tersebut dapat terlihat dari kolom ‘Employee Type’, ‘Department’, ‘Workplace Factor’, ‘Access to EAPs’, ‘Turnover Intention’. Dimana karakteristik tersebut tidak diikuti oleh kategori ‘Burnout Frequency’ yang memiliki perbedaan sekitar 12% yaitu, pada 44.4% dan 38.8%. dimana perbedaan tersebut tidak terlalu signifikan, berbeda dengan kolom seperti ‘Employee Type’, dengan perbedaan ‘hanya’ 10%, namun persentase kategori terbanyak merupakan 2x dari persentase kategori kedua terbanyak.

```python
# Filter fitur kategorikal
categorical_features = raw_df.select_dtypes(exclude=[np.number]).columns
df_categorical = raw_df[categorical_features]
df_categorical = df_categorical.drop(columns=['Employee ID'])

# Tentukan jumlah fitur kategorikal
# Update categorical_features after dropping 'Employee ID'
categorical_features = df_categorical.columns
num_cats = len(categorical_features)
num_rows = (num_cats + 1) // 2  # Maksimum 2 kolom per baris

# Buat subplot grid yang dinamis
fig, ax = plt.subplots(nrows=num_rows, ncols=2, figsize=(15, num_rows * 5))

# Jika hanya 1 baris, ubah `ax` menjadi array 2D
if num_rows == 1:
    ax = np.expand_dims(ax, axis=0)

# Loop untuk plot bar chart
for i, col in enumerate(categorical_features):
    row = i // 2
    col_idx = i % 2

    # Hitung persentase frekuensi
    value_counts = df_categorical[col].value_counts(normalize=True).mul(100).sort_values()

    # Plot bar chart
    value_counts.plot(kind='barh', ax=ax[row, col_idx], width=0.8, color='red')

    # Tambahkan teks persen di dalam bar
    for index, value in enumerate(value_counts):
        ax[row, col_idx].text(value, index, f'{value:.1f}%', fontsize=12, weight='bold', va='center')

    ax[row, col_idx].set_xlim([0, 100])
    ax[row, col_idx].set_xlabel('Frequency Percentage', fontsize=12)
    ax[row, col_idx].set_title(f'{col}', fontsize=15)

# Hapus subplot kosong yang tidak digunakan
for i in range(num_cats, num_rows * 2):
    fig.delaxes(ax.flatten()[i])

plt.suptitle('Distribution of Categorical Variables', fontsize=22)
plt.tight_layout()
plt.subplots_adjust(top=0.95)
plt.show()
```



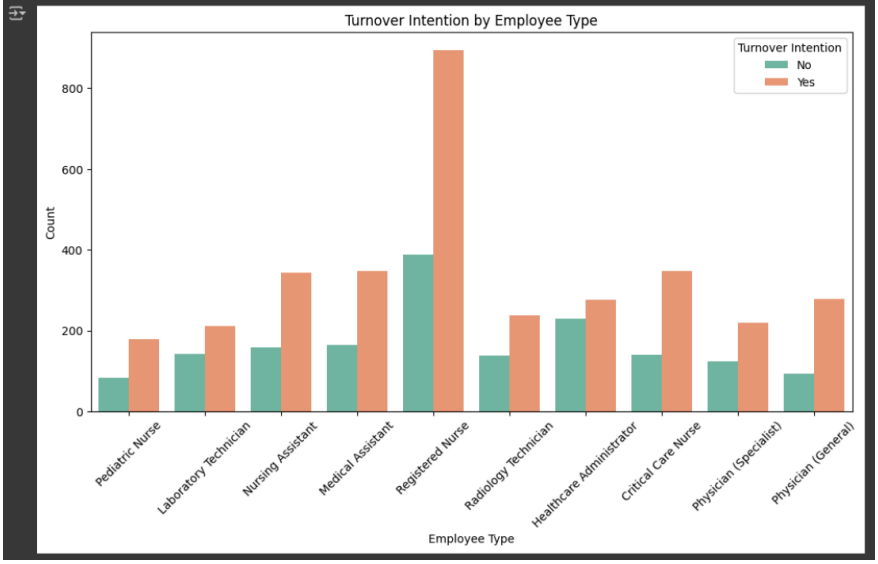
Berikutnya, kami menyajikan Heatmap korelasi dari variabel pada dataset. Dimana terlihat bahwa korelasi dari tiap variabel cenderung rendah yang berada pada nilai 0.5, atau bahkan lebih rendah lagi. Nilai-nilai tersebut dapat dikatakan kurang berkorelasi, dan kurang memiliki hubungan terhadap masing-masing kolom.

```
# Heatmap korelasi
plt.figure(figsize=(10, 8))
corr_matrix = raw_df.select_dtypes(include=['number']).corr()
sns.heatmap(corr_matrix, annot=True, cmap='coolwarm', fmt='.2f')
plt.title('Correlation Heatmap')
plt.show()
```



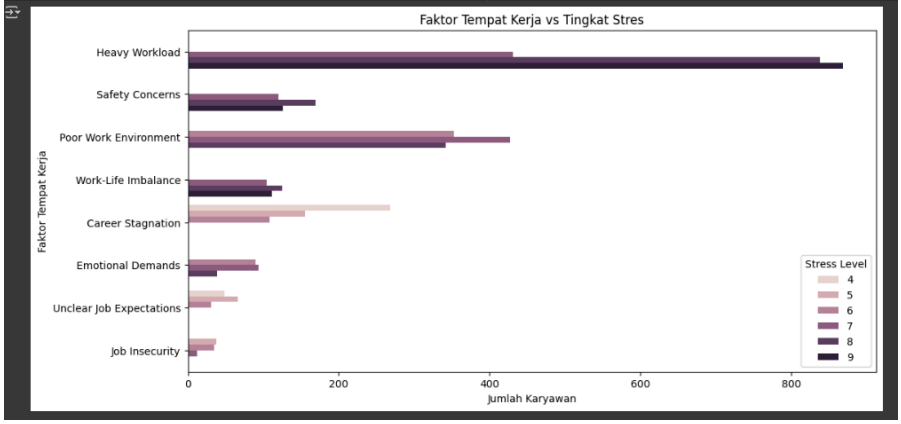
Terlihat bahwa, perbandingan antara keinginan para karyawan yang ingin berpindah profesi lebih tinggi dari pada yang tidak ingin berpindah profesi. Terlihat juga, bahwa pada kategori dengan jumlah karyawan tertinggi memiliki jumlah 'Turnover Intention' yang jauh lebih signifikan dibandingkan dengan kategori lainnya dengan jumlah yang relatif tidak terlalu banyak

```
# Visualisasi Turnover Intention dengan Employee Type
plt.figure(figsize=(12, 6))
sns.countplot(x='Employee Type', hue='Turnover Intention', data=raw_df, palette='Set2')
plt.title('Turnover Intention by Employee Type')
plt.xticks(rotation=45)
plt.xlabel('Employee Type')
plt.ylabel('Count')
plt.legend(title='Turnover Intention')
plt.show()
```



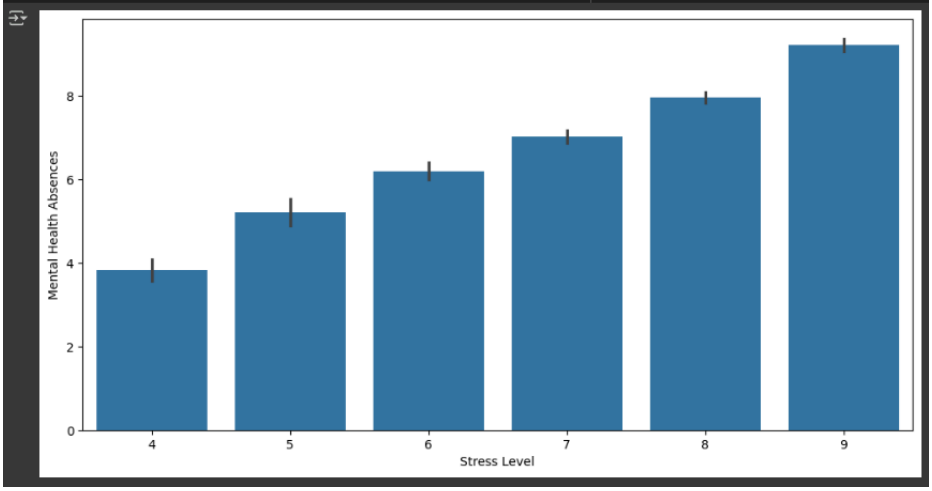
Berikutnya, dapat diobservasi bahwa faktor tempat kerja memiliki pengaruh yang cukup signifikan terhadap tingkat stress dari karyawan. Dimana terlihat bahwa faktor seperti 'Job Insecurity' tidak terlalu mengakibatkan tingkat stress yang tinggi, namun faktor seperti 'Heavy Workload' menghasilkan banyak karyawan dengan tingkat stress yang tinggi pula.

```
# Menentukan ukuran lebar dan tinggi plot
plt.figure(figsize=(12,6))

# Membuat grafik dengan "Stress Level" sebagai variabel dependen
# dan "Workplace Factor" sebagai variabel independen.
# Parameter `hue` memberikan warna berbeda untuk setiap kategori "Stress Level".
sns.countplot(y=raw_df["Workplace Factor"], hue=raw_df["Stress Level"])

# Menambahkan judul pada grafik
plt.title("Faktor Tempat Kerja vs Tingkat Stres")

# Menetapkan label untuk sumbu Y
plt.ylabel("Faktor Tempat Kerja")

# Menetapkan label untuk sumbu X
plt.xlabel("Jumlah Karyawan")

# Menampilkan grafik
plt.show()
```



Dapat dilihat juga bahwa, banyaknya cuti yang diambil oleh karyawan dengan alasan kesehatan mental memiliki hubungan yang sangat erat terhadap tingkat stress yang dialami oleh karyawan. Dimana, terlihat bahwa semakin tinggi tingkat stress yang dirasakan oleh karyawan, maka mereka cenderung mengambil lebih banyak cuti dengan alasan kesehatan mental.

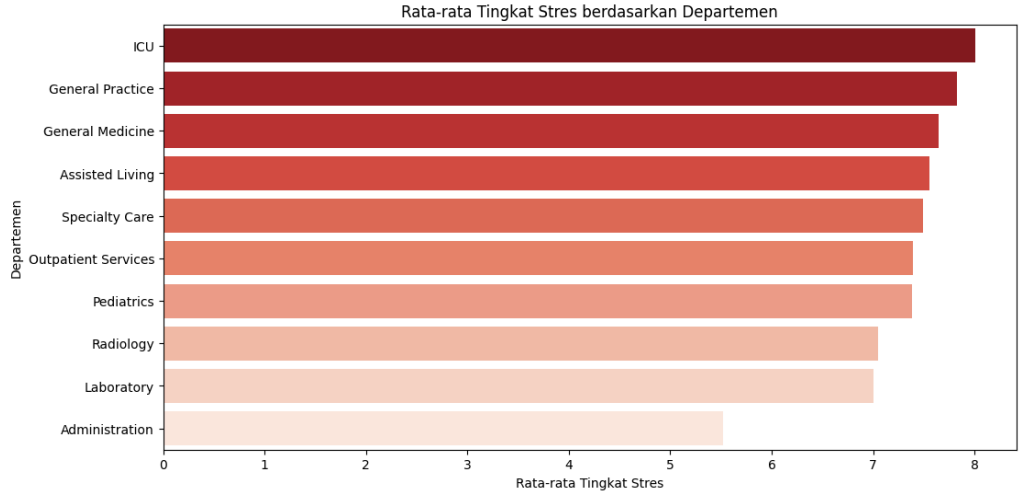
```
[17] # Menentukan ukuran lebar dan tinggi plot
     plt.figure(figsize=(12,6))

     # Membuat grafik dengan "Mental Health Absences" sebagai variabel dependen
     # dan "Stress Level" sebagai variabel independen.
     # Parameter `hue` memberikan warna berbeda untuk setiap kategori (jika ada).
     sns.barplot(x=raw_df["Stress Level"], y=raw_df["Mental Health Absences"])

     # Menampilkan grafik
     plt.show()
```



Selanjutnya, ditampilkan juga rata-rata tingkat stress yang dirasakan oleh karyawan dari masing-masing departemen. Dimana departemen yang menangani urusan darurat seperti ICU, ataupun praktisi umum memiliki tingkatan stress yang lebih tinggi. Namun, terlihat juga rata-rata tingkat stress dari hampir setiap departemen tidak memiliki perbedaan yang signifikan, dimana departemen Administrasi merasakan tingkat stress yang lebih rendah dengan perbedaan yang cukup signifikan dengan departemen lainnya.

```python
# Analisis Kelompok: Rata-rata Tingkat Stres berdasarkan Departemen
avg_stress_by_department = raw_df.groupby('Department')['Stress Level'].mean().sort_values(ascending=False)

# Menampilkan hasil perhitungan rata-rata tingkat stres
print("Rata-rata Tingkat Stres berdasarkan Departemen:\n", avg_stress_by_department)

# Menentukan palet warna
palette = sns.color_palette("Reds", len(avg_stress_by_department))[::-1]

# Menentukan ukuran lebar dan tinggi plot
plt.figure(figsize=(12,6))

# Membuat grafik batang untuk menampilkan rata-rata tingkat stres berdasarkan departemen
sns.barplot(x=avg_stress_by_department.values,
            y=avg_stress_by_department.index,
            hue=avg_stress_by_department.index,  # Menggunakan index sebagai hue
            palette=palette,
            legend=False)  # Menonaktifkan legenda

# Menambahkan judul grafik
plt.title('Rata-rata Tingkat Stres berdasarkan Departemen')

# Menetapkan label untuk sumbu X dan Y
plt.xlabel('Rata-rata Tingkat Stres')
plt.ylabel('Departemen')

# Menampilkan grafik
plt.show()
```

```
Rata-rata Tingkat Stres berdasarkan Departemen:
 Department
ICU                   8.012295
General Practice      7.822581
General Medicine      7.643024
Assisted Living       7.553785
Specialty Care        7.489796
Outpatient Services   7.393762
Pediatrics            7.385496
Radiology             7.045213
Laboratory            7.000000
Administration        5.516765
Name: Stress Level, dtype: float64
```



4. Data Preparation:

Pada tahapan Data Preparation, hal yang kami lakukan berupa feature extraction, label encoding, dan normalisasi. Pertama, kami membersihkan dataset dengan membuang kolom ‘Employee ID’ yang tidak berpengaruh terhadap tahapan pembuatan model yang mungkin akan dilakukan nantinya.

1. Encoding

```
[19] df_clean = raw_df.copy()
```

```
[20] df_clean.drop(columns=["Employee ID"], inplace = True)
     df_clean.head()
```

	Employee Type	Department	Workplace Factor	Stress Level	Burnout Frequency	Job Satisfaction	Access to EAPs	Mental Health Absences	Turnover Intention
0	Pediatric Nurse	Pediatrics	Heavy Workload	8	Often	2	Yes	6	No
1	Laboratory Technician	Laboratory	Safety Concerns	8	Often	1	Yes	12	No
2	Nursing Assistant	Assisted Living	Poor Work Environment	6	Occasionally	2	Yes	9	Yes
3	Medical Assistant	Outpatient Services	Poor Work Environment	7	Never	4	No	11	No
4	Registered Nurse	General Medicine	Work-Life Imbalance	8	Occasionally	2	Yes	7	No

Kemudian, kolom dengan tipe data kategorik, diubah menjadi angka menggunakan fungsi LabelEncoder dari library sklearn.

```
[21] from sklearn.preprocessing import LabelEncoder

     #List of Columns with Categorical Data
     Categorical_Columns = df_clean.select_dtypes(exclude=[np.number]).columns.tolist()
     #Dictionary to Store the Ecoded Data
     Label_Encoders = {}
     for col in Categorical_Columns:
         LE = LabelEncoder()
         df_clean[col] = LE.fit_transform(raw_df[col])
         Label_Encoders[col] = LE
```

```
[22] df_clean.head()
```

	Employee Type	Department	Workplace Factor	Stress Level	Burnout Frequency	Job Satisfaction	Access to EAPs	Mental Health Absences	Turnover Intention
0	5	7	2	8	2	2	1	6	0
1	2	5	5	8	2	1	1	12	0
2	4	1	4	6	1	2	1	9	1
3	3	6	4	7	0	4	0	11	0
4	9	2	7	8	1	2	1	7	0

Kemudian, dataset tersebut dinormalisasikan sehingga nilai-nilai pada setiap kolom pada dataset tersebut tidak memiliki perbedaan yang signifikan.

```
[23] from sklearn.preprocessing import RobustScaler

     cols = df_clean.select_dtypes(include=[np.number]).columns.tolist()

     scaler = RobustScaler()
     df_clean[cols] = scaler.fit_transform(df_clean[cols])
```

```
[24] df_clean.head()
```

	Employee Type	Department	Workplace Factor	Stress Level	Burnout Frequency	Job Satisfaction	Access to EAPs	Mental Health Absences	Turnover Intention
0	0.000000	1.00	0.0	0.0	1.0	0.0	0.0	-0.25	-1.0
1	-0.428571	0.50	1.5	0.0	1.0	-0.5	0.0	1.25	-1.0
2	-0.142857	-0.50	1.0	-2.0	0.0	0.0	0.0	0.50	0.0
3	-0.285714	0.75	1.0	-1.0	-1.0	1.0	-1.0	1.00	-1.0
4	0.571429	-0.25	2.5	0.0	0.0	0.0	0.0	0.00	-1.0