DSC680-Project3

November 22, 2025

[2]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import zipfile
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.ensemble import RandomForestRegressor, GradientBoostingRegressor
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score
from sklearn.preprocessing import OneHotEncoder
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline
import numpy as np

import warnings
warnings.filterwarnings("ignore")
from IPython.core.display import display, HTML
display(HTML("<style>.jp-OutputArea { overflow: visible !important; }</style>"))
```

```
<IPython.core.display.HTML object>
```

[3]:
```
# 1. Load the dataset
dataset_path = r"C:
 ↪\Users\amyha\OneDrive\Documents\DSC680\Mental_Health_and_Social_Media_Balance_Dataset.
 ↪csv"
df = pd.read_csv(dataset_path)
```

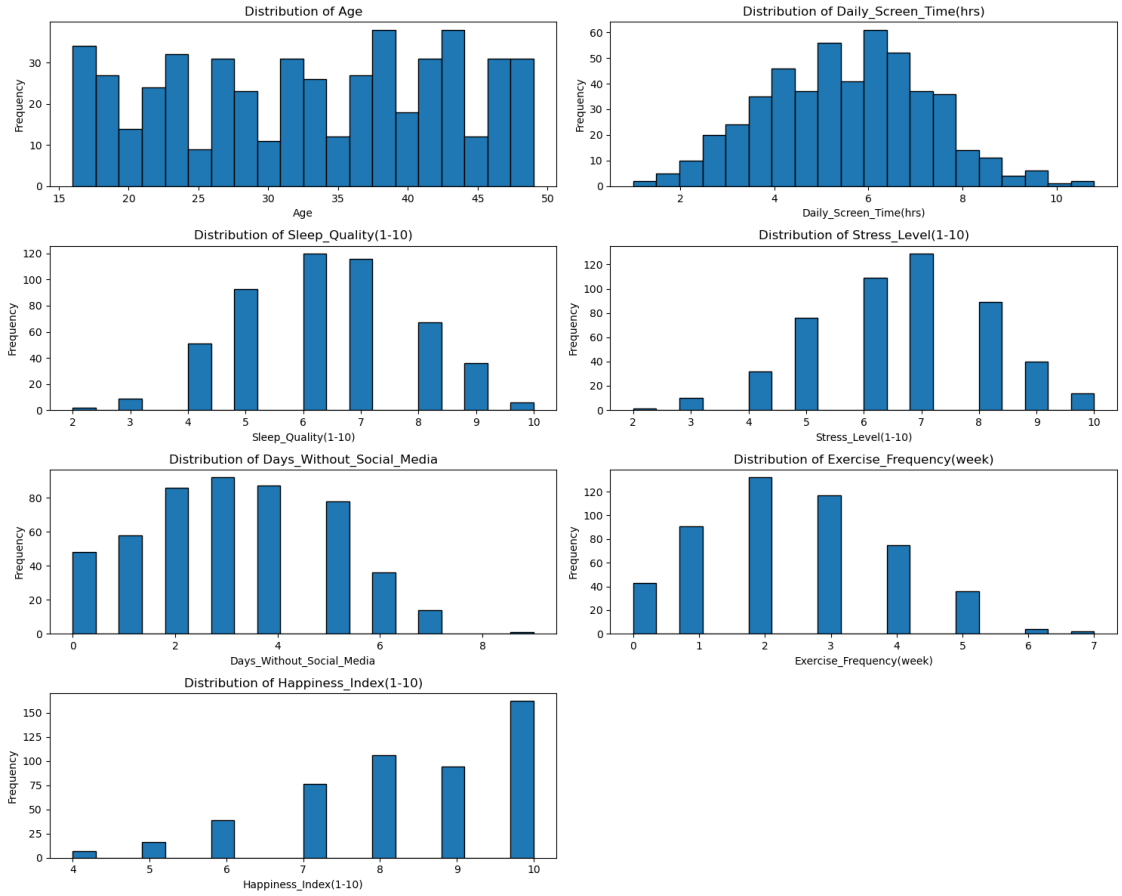
[4]:
```
# 2. Exploratory Data Analysis (EDA)
# Descriptive statistics
print(df.head())
print(df.describe())
print(df.isnull().sum())
```

```
  User_ID  Age  Gender  Daily_Screen_Time(hrs)  Sleep_Quality(1-10)  \
0    U001   44    Male                     3.1                  7.0
1    U002   30   Other                     5.1                  7.0
2    U003   23   Other                     7.4                  6.0
3    U004   36  Female                     5.7                  7.0
4    U005   34  Female                     7.0                  4.0
```

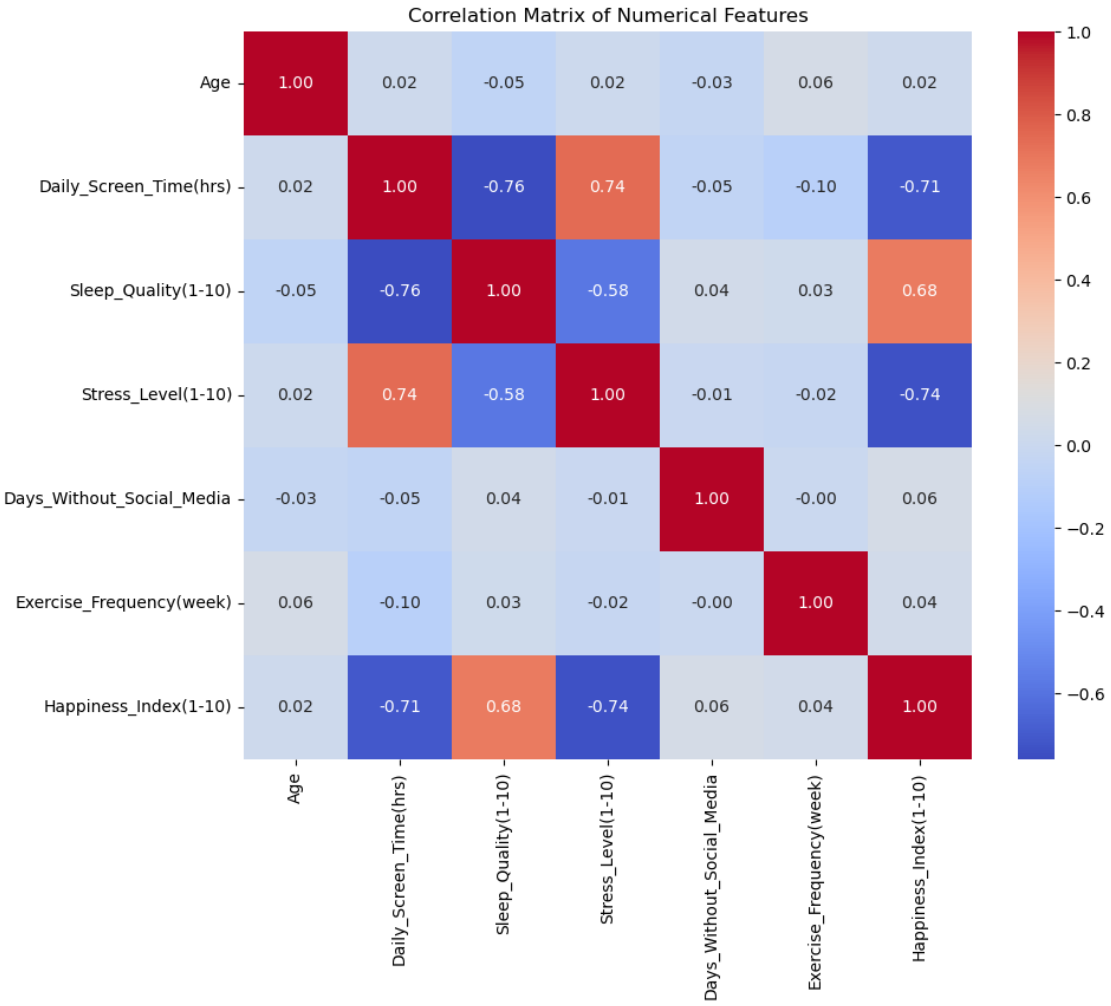
```
   Stress_Level(1-10)  Days_Without_Social_Media  Exercise_Frequency(week)  \
0                 6.0                       2.0                       5.0
1                 8.0                       5.0                       3.0
2                 7.0                       1.0                       3.0
3                 8.0                       1.0                       1.0
4                 7.0                       5.0                       1.0

  Social_Media_Platform  Happiness_Index(1-10)
0              Facebook                   10.0
1              LinkedIn                   10.0
2               YouTube                    6.0
3                TikTok                    8.0
4          X (Twitter)                     8.0
                Age  Daily_Screen_Time(hrs)  Sleep_Quality(1-10)  \
count  500.000000              500.000000           500.000000
mean    32.988000                5.530000             6.304000
std      9.960637                1.734877             1.529792
min     16.000000                1.000000             2.000000
25%     24.000000                4.300000             5.000000
50%     34.000000                5.600000             6.000000
75%     41.000000                6.700000             7.000000
max     49.000000               10.800000            10.000000

       Stress_Level(1-10)  Days_Without_Social_Media  \
count          500.000000                 500.000000
mean             6.618000                   3.134000
std              1.542996                   1.858751
min              2.000000                   0.000000
25%              6.000000                   2.000000
50%              7.000000                   3.000000
75%              8.000000                   5.000000
max             10.000000                   9.000000

       Exercise_Frequency(week)  Happiness_Index(1-10)
count                500.000000             500.000000
mean                   2.448000               8.376000
std                    1.428067               1.524228
min                    0.000000               4.000000
25%                    1.000000               7.000000
50%                    2.000000               9.000000
75%                    3.000000              10.000000
max                    7.000000              10.000000
User_ID                       0
Age                           0
Gender                        0
Daily_Screen_Time(hrs)        0
Sleep_Quality(1-10)           0
Stress_Level(1-10)            0
```

```
Days_Without_Social_Media    0
Exercise_Frequency(week)     0
Social_Media_Platform        0
Happiness_Index(1-10)        0
dtype: int64
```

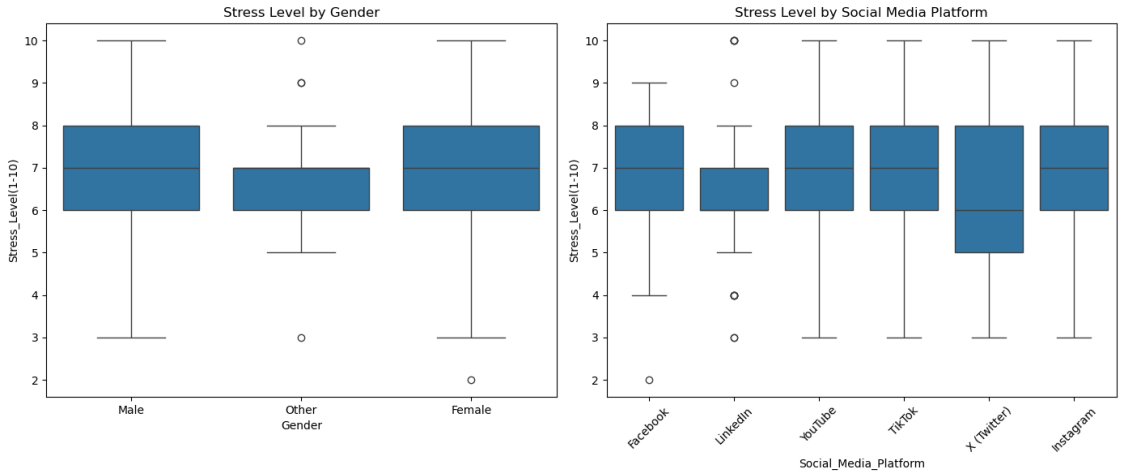
[5]:
```python
# Histograms for numeric features
num_cols = [
    "Age",
    "Daily_Screen_Time(hrs)",
    "Sleep_Quality(1-10)",
    "Stress_Level(1-10)",
    "Days_Without_Social_Media",
    "Exercise_Frequency(week)",
    "Happiness_Index(1-10)"
]

plt.figure(figsize=(15, 12))
for i, col in enumerate(num_cols):
    plt.subplot(4, 2, i + 1)
    plt.hist(df[col], bins=20, edgecolor='black')
    plt.title(f"Distribution of {col}")
    plt.xlabel(col)
    plt.ylabel("Frequency")
plt.tight_layout()
plt.show()
```



[6]:
```
# Correlation heatmap (encode categorical variables for correlation)
plt.figure(figsize=(10, 8))
corr = df[num_cols].corr()
sns.heatmap(corr, annot=True, cmap="coolwarm", fmt=".2f", square=True)
plt.title("Correlation Matrix of Numerical Features")
plt.show()
```



[7]:
```
# Box plots
plt.figure(figsize=(14, 6))
plt.subplot(1, 2, 1)
sns.boxplot(x="Gender", y="Stress_Level(1-10)", data=df)
plt.title("Stress Level by Gender")

plt.subplot(1, 2, 2)
sns.boxplot(x="Social_Media_Platform", y="Stress_Level(1-10)", data=df)
plt.xticks(rotation=45)
plt.title("Stress Level by Social Media Platform")
plt.tight_layout()
plt.show()
```



[8]:
```
X = df.drop(columns=["User_ID", "Stress_Level(1-10)"])
y = df["Stress_Level(1-10)"]

categorical_cols = ["Gender", "Social_Media_Platform"]
numerical_cols = X.drop(columns=categorical_cols).columns.tolist()

preprocessor = ColumnTransformer(transformers=[
    ("num", "passthrough", numerical_cols),
    ("cat", OneHotEncoder(drop="first"), categorical_cols)
])

models = {
    "Linear Regression": LinearRegression(),
    "Random Forest": RandomForestRegressor(random_state=42),
    "Gradient Boosting": GradientBoostingRegressor(random_state=42)
}

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

results = {}
for name, model in models.items():
    pipeline = Pipeline(steps=[
        ("preprocessor", preprocessor),
        ("model", model)
    ])
    pipeline.fit(X_train, y_train)
    y_pred = pipeline.predict(X_test)
    results[name] = {
        "MAE": mean_absolute_error(y_test, y_pred),
```

```
        "RMSE": mean_squared_error(y_test, y_pred, squared=False),
        "R2": r2_score(y_test, y_pred)
    }

results_df = pd.DataFrame(results).T.sort_values("RMSE")
print(results_df)
```

```
                          MAE      RMSE        R2
Random Forest        0.797800  0.941227  0.632159
Gradient Boosting    0.814747  0.955392  0.621004
Linear Regression    0.810950  0.966219  0.612365
```

[ ]: