Mental Health and Social Media Balance

1. Project Overview

This dataset captures the delicate relationship between social media habits and mental well-being. It combines variables such as screen time, stress level, sleep quality, digital detox days, and happiness index. Ideal for regression, correlation, or mental health prediction tasks.

2.Dataset Summary

Link - **Mental Health and Social Media Balance**

This dataset is from Kaggle. It contains Mental Health report and and social media Balance. Researchers, psychologists, and data enthusiasts can use this dataset to study how lifestyle and online activity patterns affect human emotions and overall wellness

- Rows 500
- Columns 10

- **Key features:**

User ID	-	Unique identifier for each user
Age	-	Age of users
Gender	-	Gender of Users
Daily Screen Time	-	Daily Scree time of users
Sleep Quality	-	Sleep Quality of Users
Stress Level	-	Geographic sales region
Days Without Media	-	Types of stores
Exercise	-	Exercise habits of users
Social Media	-	Social media usage of users
Happiness Index	-	Total happiness index out of 10

3. Exploratory Data Analysis using Python

<u>We began with data preparation and cleaning in python</u>

*Defining the columns

```
Index(['User_ID', 'Age', 'Gender', 'Daily_Screen_Time(hrs)',
       'Sleep_Quality(1-10)', 'Stress_Level(1-10)',
       'Days_Without_Social_Media', 'Exercise_Frequency(week)',
       'Social_Media_Platform', 'Happiness_Index(1-10)'],
      dtype='object')
```

*Initial Exploration: Used df.info () to check structure.

```
Data columns (total 10 columns):
 #   Column                     Non-Null Count  Dtype
---  ------                     --------------  -----
 0   User_ID                    500 non-null    object
 1   Age                        500 non-null    int64
 2   Gender                     500 non-null    object
 3   Daily_Screen_Time(hrs)     500 non-null    float64
 4   Sleep_Quality(1-10)        500 non-null    float64
 5   Stress_Level(1-10)         500 non-null    float64
 6   Days_Without_Social_Media  500 non-null    float64
 7   Exercise_Frequency(week)   500 non-null    float64
 8   Social_Media_Platform      500 non-null    object
 9   Happiness_Index(1-10)      500 non-null    float64
dtypes: float64(6), int64(1), object(3)
```

* Describe () for summary statistics.

	Age	Daily_Screen_Time(hrs)	Sleep_Quality(1-10)	Stress_Level(1-10)	Days_Without_Social_Media	Exercise_Frequency(week
count	500.000000	500.000000	500.000000	500.000000	500.000000	500.00000(
mean	32.988000	5.530000	6.304000	6.618000	3.134000	2.44800(
std	9.960637	1.734877	1.529792	1.542996	1.858751	1.42806
min	16.000000	1.000000	2.000000	2.000000	0.000000	0.00000(
25%	24.000000	4.300000	5.000000	6.000000	2.000000	1.00000(
50%	34.000000	5.600000	6.000000	7.000000	3.000000	2.00000(
75%	41.000000	6.700000	7.000000	8.000000	5.000000	3.00000(
max	49.000000	10.800000	10.000000	10.000000	9.000000	7.00000(

*Checking missing Data: Checked for null values.
*No missing values detected

```
: <bound method DataFrame.sum of      User_ID    Age   Gender   Daily_Screen_Time(hrs)
  0       False  False   False              False              False
  1       False  False   False              False              False
  2       False  False   False              False              False
  3       False  False   False              False              False
  4       False  False   False              False              False
  ..        ...    ...     ...                ...                ...
  495     False  False   False              False              False
  496     False  False   False              False              False
  497     False  False   False              False              False
  498     False  False   False              False              False
  499     False  False   False              False              False
```

*Head (5) for printing initial first 5 columns

	User_ID	Age	Gender	Daily_Screen_Time(hrs)	Sleep_Quality(1-10)	Stress_Level(1-10)	Days_Without_Social_Media
0	U001	44	Male	3.1	7.0	6.0	2.0
1	U002	30	Other	5.1	7.0	8.0	5.0
2	U003	23	Other	7.4	6.0	7.0	1.0
3	U004	36	Female	5.7	7.0	8.0	1.0
4	U005	34	Female	7.0	4.0	7.0	5.0

4. Predicting Happiness Using Linear Regression

- Importing required features for predicting Linear regression.
- Reading adidas CSV file and assign X and Y variables.
- Instantiates a Standard Scaler to standardize the features.
- Scales X and splits it into train/test sets (80 % train, 20 % test) .
- Random State = 42.

```
x = numeric_features.drop('Happiness_Index(1-10)', axis=1)
y = numeric_features['Happiness_Index(1-10)']

x_train, x_test, y_train, y_test = train_test_split(
    x, y, test_size=0.3, random_state=42
```

- Predicts on the test set (y pred).
- Computes and prints the Mean Squared Error and R^2 score.
- Prints the regression coefficients.

```
y_pred = model.predict(x_test)
r2 = r2_score(y_test, y_pred)
rmse = np.sqrt(mean_squared_error(y_test, y_pred))

print("\nModel Performance:")
print("R2 score:", rmse)
```

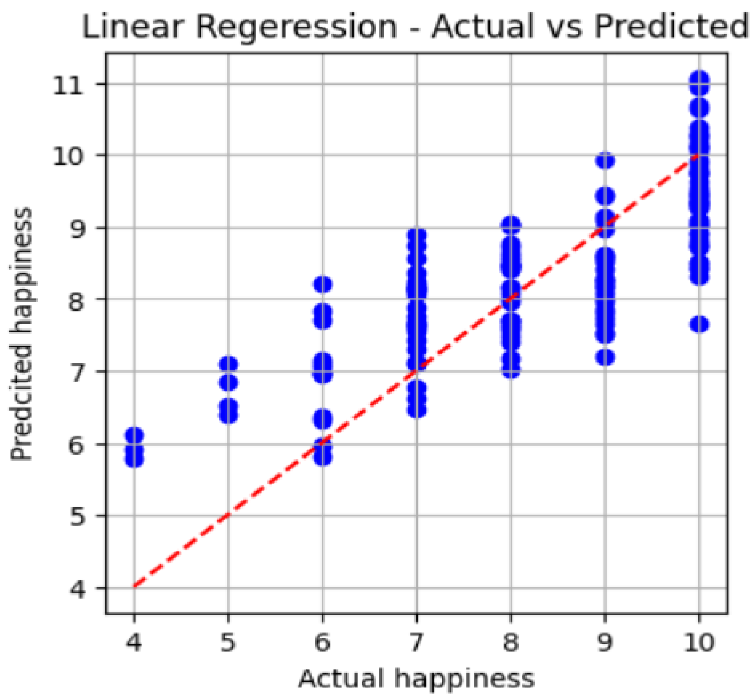
```
\Model Coefficients: [ 0.00622924 -0.07675173  0.34520363
Model Intercept: 9.33874689962726

Model Performance:
R2 score: 0.9579548353228006
```

*** The model has excellent predictive performance with an R^2 of 0.96**
***Meaning it fits the data very well.**
***The intercept (9.3387) is the baseline prediction features are zero**

- Comparing the Actual VS Predicted.

```
Actual vs Predicted:
     Actual  Predicted
361     7.0   8.110613
73      8.0   8.471536
374     9.0   7.869610
155    10.0   9.273720
104     9.0   8.004732
..      ...        ...
266    10.0  10.392033
23      7.0   7.095210
222     8.0   9.051211
261    10.0   9.464902
426     8.0   7.706504
```

```
plt.figure(figsize=(4,4))
plt.scatter(y_test, y_pred, color="blue")
plt.plot([y_test.min(), y_test.max()], [y_test.min(), y_test.max()], color="red", linestyle="--")
plt.xlabel("Actual happiness")
plt.ylabel("Predcited happiness")
plt.title("Linear Regeression - Actual vs Predicted")
plt.grid(True)
plt.show()
```



***This plot shows a linear regression comparison.**
*** Between actual happiness and predicted happiness.**
***Blue dot represent individual observations.**
***The model generally captures the positive trend.**
***The spread of points suggests moderate residual variance.**
***The model isn't perfect but gives a reasonable approximation.**

4. Predicting Stress Level Using Lasso Regression

- Importing required features for predicting Lasoo regression.
- Reading CSV file and assign X and Y variables.
- Instantiates a Standard Scaler to standardize the features.
- Scales X and splits it into train/test sets (80 % train, 20 % test).
- Random State = 42.

```
df = pd.read_csv("C:/Users/appuv/Desktop/PROJECT/Mental_Health_and_Social_Media_Balance_Dataset.csv")
numeric_features = df.select_dtypes(include=[np.number])
x = numeric_features.drop('Stress_Level(1-10)', axis=1)
y = numeric_features['Stress_Level(1-10)']

x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2, random_state=42)
scaler = StandardScaler()
x_scaled = scaler.fit_transform(x_train)
x_test_scaled = scaler.fit_transform(x_test)
```

- Create 50 values of alpha logarithmically.
- LassoCV performs with built-in cross-validation.
- The optimal alpha is retrieved from the alpha-attribute.
- A new lasso model is instantiated with the best alpha.

```
alphas = np.logspace(-3, 1, 50)
lasso_cv = LassoCV(alphas=alphas, cv=5, max_iter=10000)
lasso_cv.fit(x_train, y_train)

best_alpha = lasso_cv.alpha_
print(f"Best alpha from Cv: {best_alpha:.4f}")

lasso = Lasso(alpha=best_alpha)
lasso.fit(x_train, y_train)
```

- Print R2 and MSE

```
Best alpha from Cv: 0.0026
MSE: 0.9044380661404757
R2 score: 0.6244651776530161
```

***Best Alpha CV is 0.0026 the optimal regularization parameter is selected.**

***The Mean Squared Error measures the average squared difference between Predicted and actual values. Lower MSE indicates better fit.**

***The R^2 is 0.62 indicates how much variance in the dependent variable is explained.**

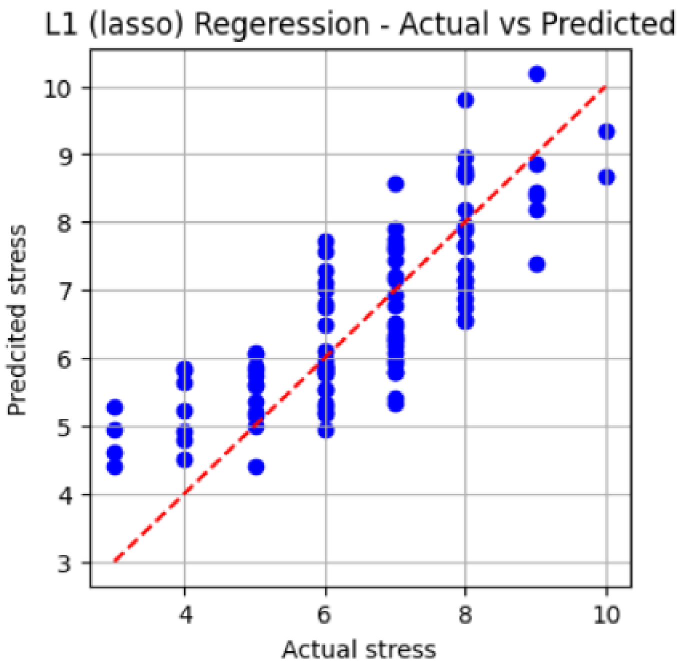
- Comparing the Actual VS Predicted.

```
Actual vs Predicted:
     Actual  Predicted
361     7.0   7.673511
73      6.0   6.490530
374     7.0   6.507964
155     5.0   5.876924
104     6.0   6.756837
..      ...        ...
347     8.0   7.957511
86      4.0   4.513907
75      5.0   5.211744
438     6.0   7.016593
15      6.0   5.852412
```

***The predicted values are continuous decimals, suggesting a regression or probabilistic model.**

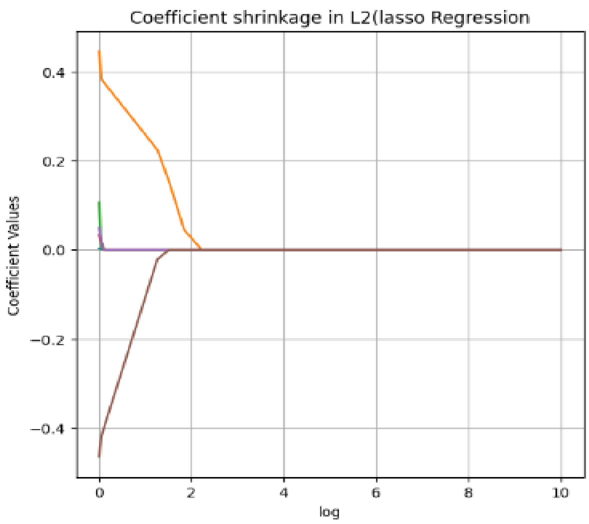
***Difference between actual and predicted indicate the model's error for each instance.**

Plotting the Actual VS Predicted



*This plot shows a lasso regression comparison.
* Between actual stress and predicted stress
*Blue dot represent individual observations.
*The model generally captures the positive trend.
*The spread of points suggests moderate residual variance.

- Coefficient Shrinkage



*X-axis represent the log of the regularization parameter.
*As log increases the penalty on the coefficient gets stronger
*Y-axis shows how the regression coefficients change with log.
*The plot helps choose an optional log by balance bias-variance
*In ridge regression, coefficients are shrunk but not set exactly to zero unlike lasso which performs variable selection.

5. CONCLUSION

- The Linear regression predicts Happiness accuracy of 96%.
- Lasso Regression gives an accuracy for Stress Level of 62%.
- Happiness is heavily driven by Sleep Quality and Stress Level.