greatlearning

PGPDSE FT Capstone Project –Final Report

Dry Eye Disease Prediction

Industry Review:

- **Industry Review – Current practices, Background Research**
    - This data from diverse subjects with age ranging from 18 to 45, enabling researchers and healthcare professionals to explore correlations between lifestyle factors and ocular health.
    - The dataset can be used for machine learning models, statistical analysis, and clinical decision-making to enhance early detection and personalized treatment strategies for DED.
- **Literature Survey - Publications, Application, past and undergoing research**
    - Data is actually got from Kaggle.
    - Person who uploaded the data is Daksh Nagra.
    - The data can also be used to predict another severe sleep related diseases such as insomnia which can be directly linked with ocular surface diseases (OSD).

Dataset and Domain:

1. Data Dictionary

In this data we have a total of **26 columns** and the dictionary or overview of the data is a below.
Output:

Columns	Description	Datatype	Units
Gender	It is a categorical column consisting of both genders	object	NA
Age	A numerical attribute with value of age of subject	int64	years
Sleep duration	It is a numerical measure of time of sleep in hours	float64	hours
Sleep quality	Ranging between 1 to 5, it provides information of quality of sleep	int64	NA
Stress level	On a scale of 1 to 5, it denotes stress levels of subject	int64	NA
Blood pressure	The combinational attribute comprising of systolic and diastolic blood pressure	object	mmHg
Heart rate	It is a measure of pulse of subject in beats per minute	int64	bpm
Daily steps	The step count of subject on a daily basis in 1000 per day units	int64	steps
Physical activity	It represents time of any kind of physical activity in minutes per day	int64	minutes
Height	Height of subject measured in cm	int64	cm
Weight	Weight of subject measured in kg	int64	kg
Sleep disorder	It helps to identify any kind of sleep disorder on case subject is aware	object	NA
Wake up during night	A categorical field to check whether subject wakes at night	object	times
Feel sleepy during day	Feel sleepy during day	object	NA
Caffeine consumption	Another category based column representing drowsiness during day	object	mg
Alcohol consumption	This field identifies caffeine consumption of subject	object	drinks
Smoking	It denotes the alcohol consumption of subject in yes-no format	object	NA
Medical issue	This categorical column depicts the smoking habits of subject	object	NA
Ongoing medication	Any kind of medical issue subject is dealing with	object	NA
Smart device before bed	The current medical prescription or medication of any kind of disease	object	minutes
Average screen time	It helps to identify the recent activity of user before sleeping	float64	hours
Blue-light filter	The measure of screen time of subject on an average in a day	object	NA
Discomfort Eye-strain	This attribute contributes to eye protection while exposure to screen	object	NA
Redness in eye	A categorical field if subject feels any kind of strain or discomfort in eye	object	NA
Itchiness/Irritation in eye	It is used to identify redness in eye	object	NA
Dry Eye Disease	To categorize prevalence of itchiness in eyes, this field is used	object	NA

2. Variable categorization (count of numeric and categorical)

There are totally 10 numerical columns and 16 categorical columns

Output:

```
# Count of numerical and categorical variables
numeric = df.select_dtypes(include= 'number').columns.to_list()
catagoric = df.select_dtypes(include= 'object').columns.to_list()
len(numeric),len(catagoric)
```

```
(10, 16)
```

3. **Pre Processing Data Analysis or checking the defects (count of missing/ null values, redundant columns, etc.**
    1. **Duplicates in Data**

    There are no Duplicates or duplicate values in data

    Output:

Checking for duplicate values

```
df.duplicated().sum()
```

```
0
```

    2. **Missing values**

    There are no missing values in data

    Output:

Checking for missing values ¶

```
# Missing Values
missing_values = df.isnull().sum()
missing_percentage = (missing_values / len(df)) * 100
print('Count of missing values ',missing_values.sum())
print("\nMissing Value Percentage:")
print(missing_percentage)
```

```
Count of missing values  0
```

    3. **Statistical Summary**

    For numerical columns

    The statistical summary for **11 numerical** are is also known **as 5-point summary.** Below we have the inferences from descriptive analysis

    Output:

	Age	Sleep duration	Sleep quality	Stress level	Heart rate	Daily steps	Physical activity	Height	Weight	Average screen time
count	20000.000000	20000.000000	20000.000000	20000.000000	20000.000000	20000.000000	20000.000000	20000.000000	20000.000000	20000.000000
mean	31.422800	6.998245	2.997250	2.993750	79.912200	10536.900000	90.069750	174.865900	74.891850	5.519885
std	8.103717	1.731723	1.412283	1.407235	11.808279	5752.729186	52.317283	14.719903	14.733839	2.606305
min	18.000000	4.000000	1.000000	1.000000	60.000000	1000.000000	0.000000	150.000000	50.000000	1.000000
25%	24.000000	5.500000	2.000000	2.000000	70.000000	6000.000000	45.000000	162.000000	62.000000	3.300000
50%	31.000000	7.000000	3.000000	3.000000	80.000000	11000.000000	91.000000	175.000000	75.000000	5.500000
75%	39.000000	8.500000	4.000000	4.000000	90.000000	16000.000000	135.000000	188.000000	88.000000	7.800000
max	45.000000	10.000000	5.000000	5.000000	100.000000	20000.000000	180.000000	200.000000	100.000000	10.000000

    Inferences:

1. **Age**: This is a relatively young to middle-aged population. The narrow age range may affect generalizability to older adults or children.
2. **Sleep duration**: On average, participants meet recommended sleep duration (7–9 hours). However, a significant portion sleeps below that threshold (min = 4), which may affect health and screen-time related disorders.
3. **Sleep quality:** Normal distribution centered at moderate sleep quality. There's noticeable variance, suggesting some individuals experience poor sleep regularly.
4. **Stress level**: stress levels average around medium, with enough spread to detect differences among individuals. This could be a critical feature when analyzing lifestyle-related conditions.
5. **Heart rate**: All within normal resting heart rate range.
6. **Daily steps:** Quite active on average — the general guideline is 10,000 steps/day. High variance suggests significant lifestyle differences (sedentary vs. active users).
7. **Physical activity**: On average, people meet the recommended 30 mins/day. But some are completely inactive (0 minutes) — a possible risk factor for lifestyle diseases.
8. **Height:** Represents a fairly typical adult height distribution.
9. **Weight:** Standard weight range for adults.
10. **Average Screen Time**: High screen time on average — this may correlate strongly with eye strain, dry eye disease (DED), and sleep quality issues. A key feature

For Categoric columns:

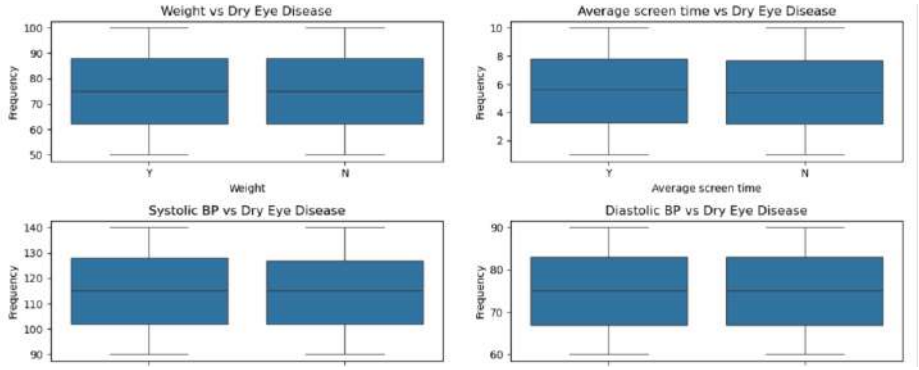
The statistical analysis for 18 categoric columns is 4 points summary – count, unique, top, freq,

	count	unique	top	freq
Gender	20000	2	M	10028
Blood pressure	20000	1581	109/73	27
Sleep disorder	20000	2	N	10069
Wake up during night	20000	2	N	10000
Feel sleepy during day	20000	2	N	10178
Caffeine consumption	20000	2	Y	10089
Alcohol consumption	20000	2	Y	10009
Smoking	20000	2	N	10017
Medical issue	20000	2	N	10111

Mostly, all columns are having unique values but **blood pressure** looks like having anomaly issue.

4. Outliers

From plotting all the boxplot graphs for the numerical columns, we can conclude there is no outliers present in the data.



- No serve outliers in most features
- Distributions appear fairly normal or slightly skewed in some cases.

5. Defects in Data

We have to check for the anomalies present in the data.

From the descriptive analysis of categorical data, we can observe that **blood pressure** has anomaly issue.

6. Alternate sources of data that can supplement the core dataset (at least 2-3 columns) (Feature Engineering)

We are not having any additional source of data but created few columns from the existing data -

1. For **Systolic BP** and **Diastolic BP** –

```
# We will Split 'Blood pressure' into 'Systolic' and 'Diastolic'
df[['Systolic BP', 'Diastolic BP']] = df['Blood pressure'].str.split('/', expand=True)

# Convert the new columns to numeric
df['Systolic BP'] = pd.to_numeric(df['Systolic BP'], errors='coerce')
df['Diastolic BP'] = pd.to_numeric(df['Diastolic BP'], errors='coerce')

# Drop the original 'Blood pressure' column
df = df.drop(columns=['Blood pressure'])
```

Also, we have classified three more columns from the existing data.

2. **BP_category** – This is done using from Systolic BP and Diastolic BP is done based on American Heart Association

BLOOD PRESSURE CATEGORY	SYSTOLIC mm Hg (upper number)	and/or	DIASTOLIC mm Hg (lower number)
NORMAL	LESS THAN 120	and	LESS THAN 80
ELEVATED	120 – 129	and	LESS THAN 80
HIGH BLOOD PRESSURE (HYPERTENSION) STAGE 1	130 – 139	or	80 – 89
HIGH BLOOD PRESSURE (HYPERTENSION) STAGE 2	140 OR HIGHER	or	90 OR HIGHER
HYPERTENSIVE CRISIS (consult your doctor immediately)	HIGHER THAN 180	and/or	HIGHER THAN 120

3. **Sleep_Category** – This is done using Sleep duration column is done on basis data from WHO.

According to National Heart Lung and Blood Institute of America, Experts recommend that Adult should have sleep at least 7-9 hours of sleep every day.

4. **Screen Time Category** – This is done using Average Screen Time column on the basis of National Health institute
5. **BMI**: This column is found using the columns Height and Weight.
6. **Pusle Pressure**: This is done by the difference between Systolic and Diastolic BPs

7. **Project Justification - Project Statement, Complexity involved, Project Outcome –**

Project Justification:

1. **Project Statement:**

   This dataset is used for prediction of dry diseases on key attributes like sleep quality, sleep duration, eye redness, itchiness, screen time, blue-light filter usage and eye strain for the people under the age category of 18-45 in a given population.

2. **Project Outcome:**

   We classify people who are having the problem of Dry Eye Disease so that they can get their eyes treated at the earliest.

3. **Complexity Involved:**

   The Complexity issues which we face are –

   - Class imbalance
   - Skewed distribution
   - Anomalies
   - Outliers

8. **Commercial, Academic or Social value**

The **Social value** in this problem we are helping people **to improve their eyesight level** and have health lifestyle as without proper vision as they may face problems in day-to-day life.

Data Exploration (EDA)

1. Relationship between variables

We have found the correlation between all variables present in the data.

[70]:

	Age	Sleep duration	Sleep quality	Stress level	Heart rate	Daily steps	Physical activity	Height	Weight	Average screen time	Systolic BP	Diastolic BP
Age	1.000000	0.004857	0.002513	0.008379	-0.001196	0.001302	-0.009191	-0.005171	0.003908	0.003177	0.018157	-0.016013
Sleep duration	0.004857	1.000000	-0.006892	-0.006088	-0.029175	0.002823	0.001858	0.005259	0.000222	-0.004208	-0.000939	-0.000648
Sleep quality	0.002513	-0.006892	1.000000	0.000721	-0.014326	-0.003074	-0.010329	0.009380	0.005604	0.004697	0.007444	-0.006604
Stress level	0.008379	-0.006088	0.000721	1.000000	-0.008332	-0.005978	0.004272	-0.000651	-0.000611	-0.001344	0.003786	0.004450
Heart rate	-0.001196	-0.029175	-0.014326	-0.008332	1.000000	-0.001899	0.001334	-0.005229	-0.009639	0.002467	-0.005864	0.004728
Daily steps	0.001302	0.002823	-0.003074	-0.005978	-0.001899	1.000000	0.008413	-0.016801	-0.000619	-0.008670	-0.003174	-0.001200
Physical activity	-0.009191	0.001858	-0.010329	0.004272	0.001334	0.008413	1.000000	-0.005989	0.016160	0.006469	-0.003794	0.006379
Height	-0.005171	0.005259	0.009380	-0.000651	-0.005229	-0.016801	-0.005989	1.000000	-0.000974	0.012817	-0.007917	0.001422
Weight	0.003908	0.000222	0.005604	-0.000611	-0.009639	-0.000619	0.016160	-0.000974	1.000000	0.007798	0.003181	-0.003476
Average screen time	0.003177	-0.004208	0.004697	-0.001344	0.002467	-0.008670	0.006469	0.012817	0.007798	1.000000	-0.009392	-0.009701
Systolic BP	0.018157	-0.000939	0.007444	0.003786	-0.005864	-0.003174	-0.003794	-0.007917	0.003181	-0.009392	1.000000	-0.000374
Diastolic BP	-0.016013	-0.000648	-0.006604	0.004450	0.004728	-0.001200	0.006379	0.001422	-0.003476	-0.009701	-0.000374	1.000000

From the above code we can see that,

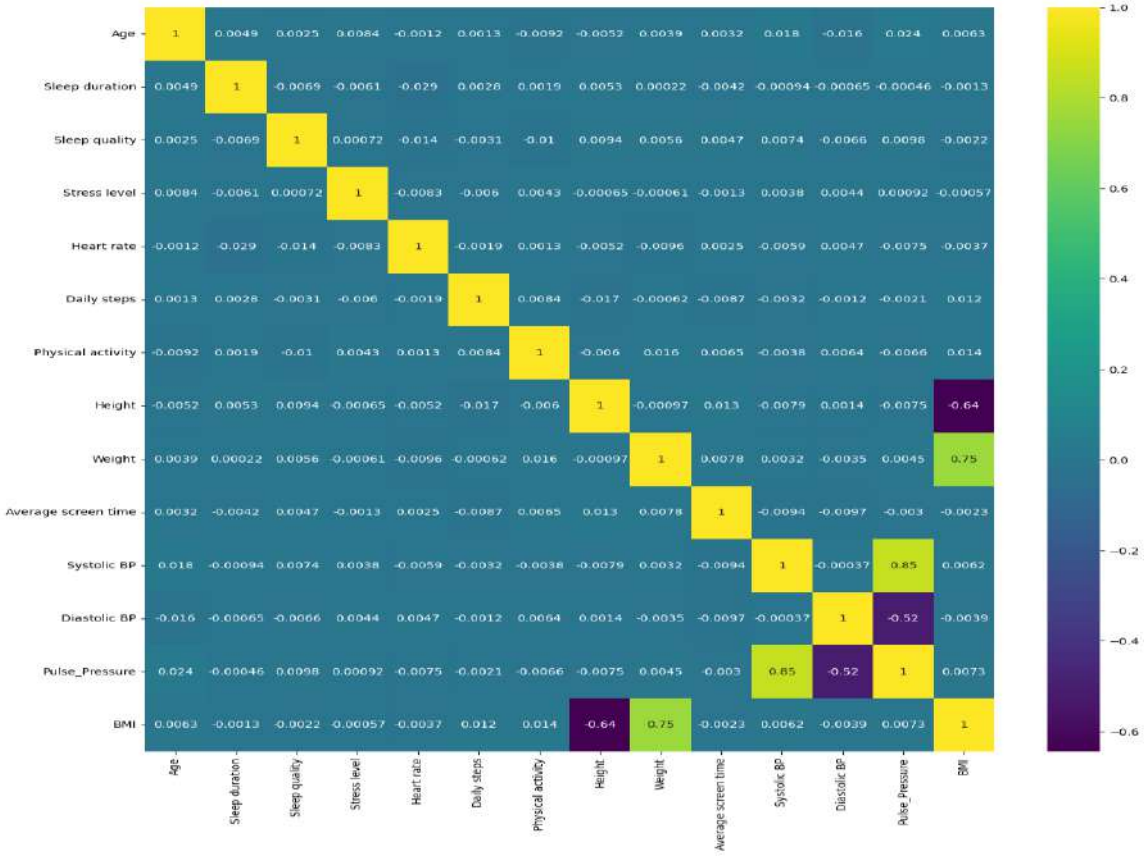
a) Age has a **weak positive** correlation with Sleep duration, Sleep quality, Stress level, Weight, Average screen time, Daily steps, and Systolic BP, and a **weak negative** correlation with Heart Rate, Physical Activity, Diastolic BP, and Height.
b) Sleep duration has a **weak positive** correlation with Daily steps and Physical activity, and a **weak negative** correlation with Sleep quality, Stress level, Heart rate, Average screen time, Systolic BP, and Diastolic BP.
c) Sleep quality has a **weak positive** correlation with Stress level, Systolic BP, and a **weak negative** correlation with Sleep duration, Heart rate, Daily steps, and Physical activity.
d) Sleep level has a **weak positive** correlation with Sleep quality, Heart rate, and Diastolic BP, and a **weak negative** correlation with Sleep duration, Daily steps, Physical activity, Height, and Weight.
e) Heart Rate has a **weak positive** correlation with Stress level and Diastolic BP, and a **weak negative** correlation with Age, Sleep duration, Sleep quality, Daily steps, Physical activity, Height, and Weight.
f) Daily Steps has a **weak positive** correlation with Sleep duration and Physical activity, and a **negative** correlation with Age, Sleep quality, Stress level, Heart rate, Height, Weight, and Average screen time.
g) Physical Activity has a **weak positive** correlation with Sleep duration and Daily steps, and a **weak negative** correlation with Age, Sleep quality, Stress level, Heart rate, Height, and Average screen time.
h) Height has a **weak positive** correlation with Average Screen Time, and a **weak negative** correlation with Age, Sleep duration, Stress level, Heart rate, Daily steps, and Physical activity.
i) Weight has a **weak positive** correlation with Age and Average screen time, and a **weak negative** correlation with Stress level, Heart rate, Daily steps, and Height.
j) Average Screen Time has a **positive correlation** with Age, Height, and Weight, and a **negative correlation** with Sleep duration and Daily steps.
k) Systolic BP has a **weak positive** correlation with Sleep Quality, and a **weak negative** correlation with Sleep duration, Height, and Average screen time.
l) Diastolic BP has a **weak positive** correlation with Stress level and Heart rate, and a **weak negative** correlation with Sleep duration and Average screen time.

2. Check for

a. Multi-collinearity:

Below is the diagram for multicollinearity:

Output:



From the above output we can observe that,

Columns – Height, Weight, Systolic_BP, Diastolic_BP , Pulse Pressure, BMI are having **multicollinearity** issues**.** Also,

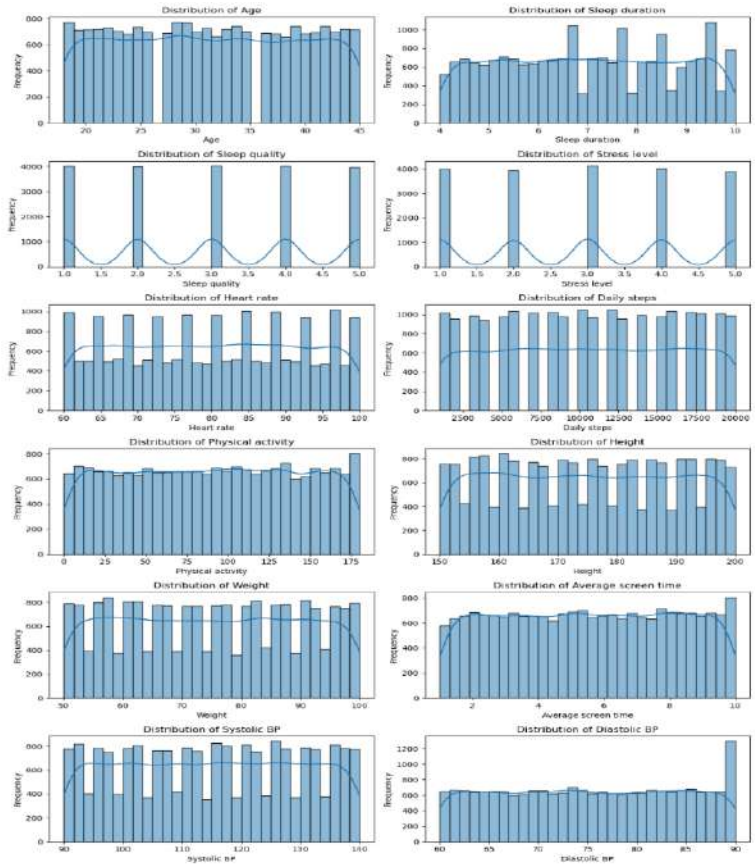
a. Systolic BP shows a **strong positive** correlation with Pulse Pressure (0.85), meaning higher systolic pressure is associated with higher pulse pressure.
b. BMI has a **strong positive** correlation with Weight (0.75) and a strong negative correlation with Height (-0.64), which is expected given BMI's formula.
c. Diastolic BP is **moderately negatively** correlated with Pulse Pressure (-0.52), indicating an inverse relationship.
d. Most features like age, sleep duration, physical activity, screen time, etc., show very **weak or negligible** correlations with each other (values close to 0), suggesting they are largely independent in this dataset.

b. Distribution of variables

We have done analysis of the distribution of the variables in both univariant and bivariant methods.

Univariant Analysis:

Output:

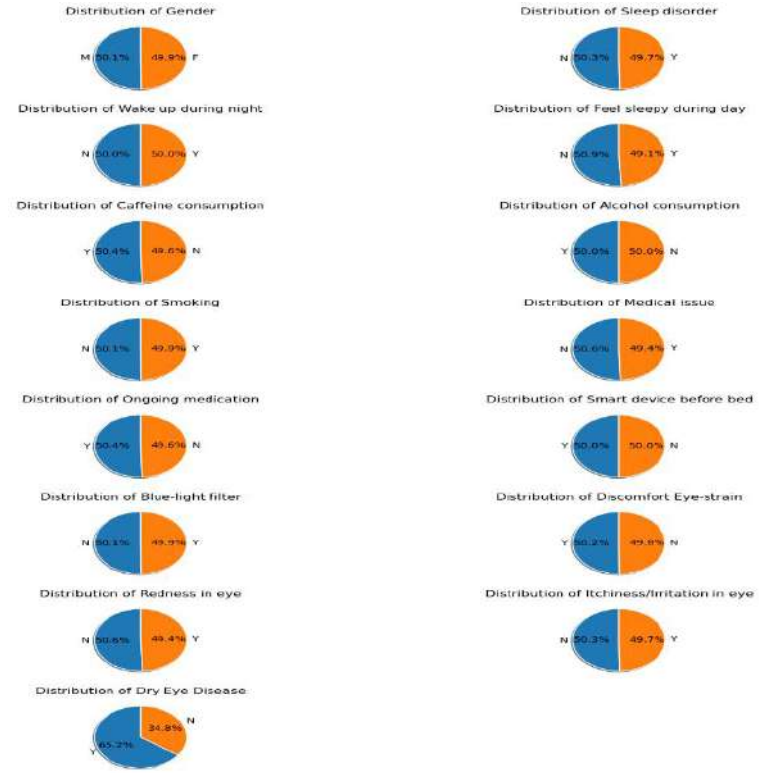


Inference:

c. **Age**: The age distribution appears roughly uniform between 18 and 45, but there's a notable absence of older adults (above 45 years).
d. **Sleep duration**: Sleep Duration is slightly right-skewed — many sleep around 6–8 hours, which is ideal, but a subset sleeps <6 hours.
e. **Sleep quality**: Sleep Quality shows an even spread — indicating variability, and low sleep quality is a known DED risk factor.
f. **Stress level**: Stress Level: Spread across the full 1–5 scale. Some users report high stress.
g. **Average Screen Time**: Nearly uniform, with many individuals having >6 hours/day of screen time. High screen exposure reduces blink rate, a direct trigger for Dry Eye Disease — this is likely a strong predictive feature.
h. **Heart rate**: Uniform, but some individuals are on the higher side (90–100 bpm).

Categorical:

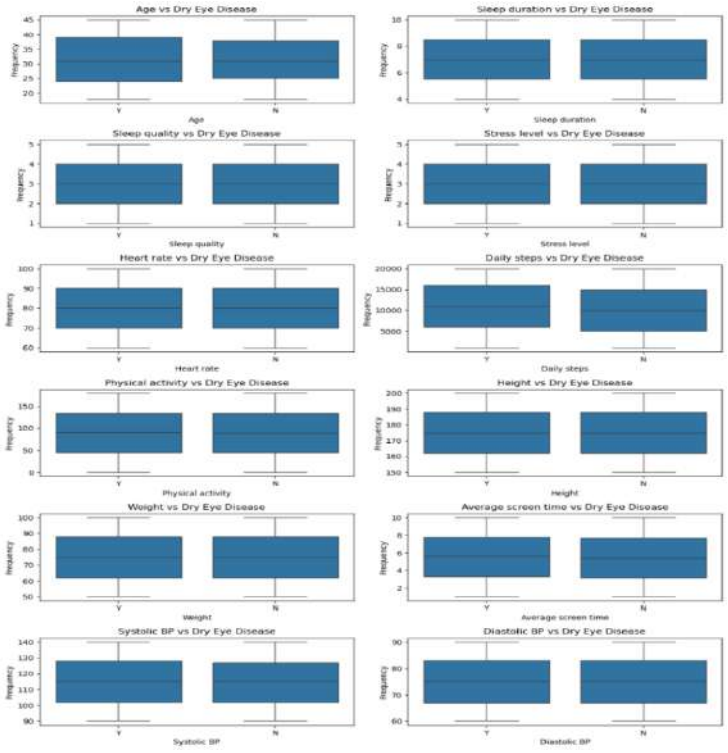
Output:



Inference:

- **Dry Eye Disease**: Yes: 65.2%, No: 34.8%
- Indicates a mild class imbalance — might influence classification metrics like accuracy.

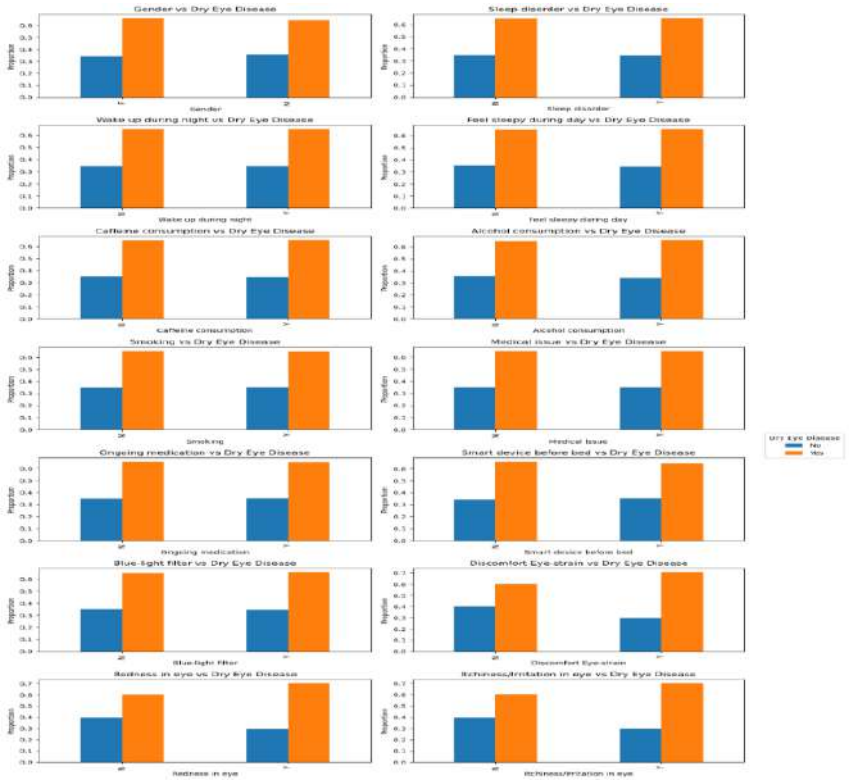
Bivariant Analysis:



Inference:

3. **Age**: Slight upward trend in DED frequency with increasing age. Indicates that older individuals are more prone to Dry Eye Disease.
4. **Sleep Duration**: Moderate fluctuations, but overall, DED frequency seems slightly lower with longer sleep durations. Poor sleep may be linked to increased risk of DED.
5. **Heart Rate**: No strong pattern observed, though slightly more DED cases are seen at lower and higher extremes, suggesting possible impact of health/stress levels.
6. **Daily Steps**: Slight downward trend – more physically active individuals (higher steps) seem to have fewer DED cases. Physical activity may help reduce risk.
7. **Physical Activity**: High variability, but generally lower DED frequency at moderate-to-high activity levels. Reinforces that inactivity might correlate with DED.
8. **Height**: No consistent relationship with DED observed. Likely not a significant predictor.
9. **Weight**: Slight upward trend in DED frequency with increasing weight.
10. **Average Screen Time**: Clear upward trend: more screen time strongly correlates with higher DED frequency. Indicates screen exposure is a major risk factor.
11. **Systolic BP**: Moderate rise in DED frequency at higher systolic BP. High BP might be indirectly linked to DED via overall health conditions.
12. **Diastolic BP**: Similar trend to systolic – DED increases with higher diastolic BP. Suggests possible vascular or systemic health impact on eye health.

Categorical:



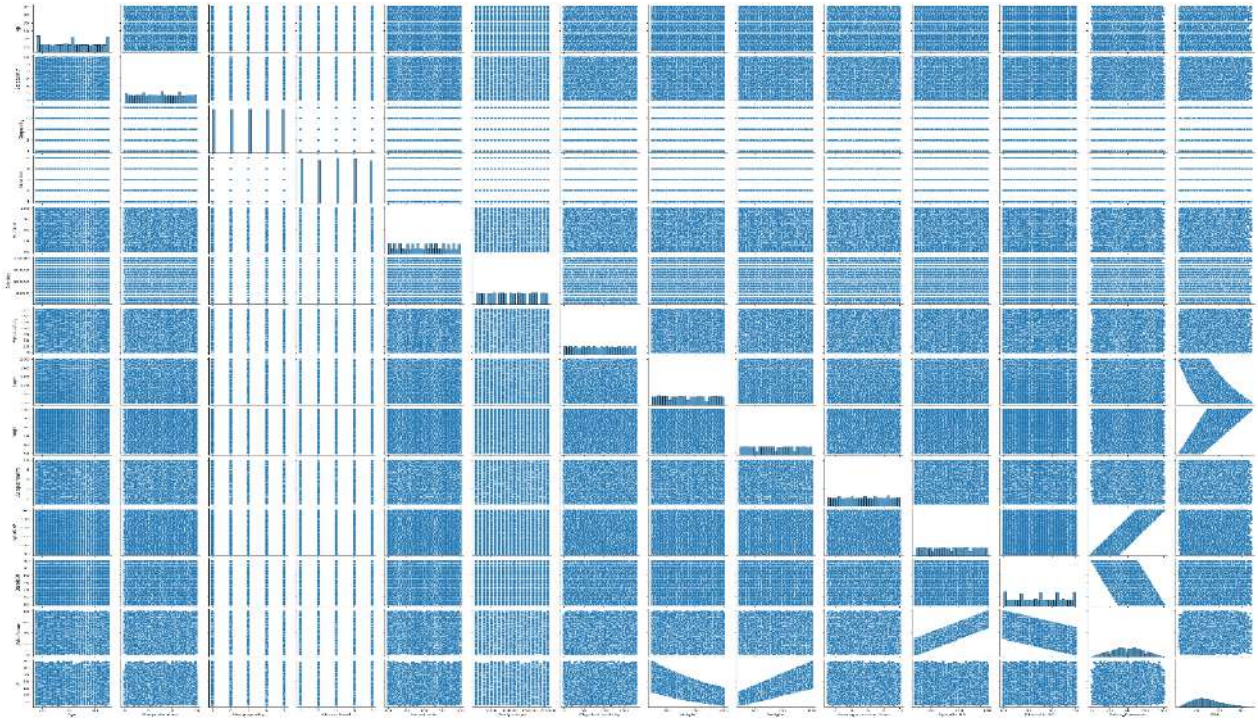
Inference:

a. **Females**, individuals with **poor sleep quality, high stress, and sleep disorders** are more likely to have **Dry Eye Disease (DED).**

b. Excessive **screen time**, especially before bed, and symptoms like **eye strain, redness, and irritation** are strong indicators of **DED**.
c. Use of **blue-light filters** and maintaining good sleep hygiene may help reduce the risk of **DED**.
d. Health factors such as **medical issues, ongoing medication, and smoking** also show moderate influence on **DED** presence.

Mutli variant Analysis:

1. We can have plotted a pair plot to show the combination of each variables over the other columns and plotted it.
   Output:



e. Statistical significance of variables or Relationship between variables.

We have check how all the independent variables (all column expect Dry Eye Disease) are statistically significant with the target variable (Dry Disease Eye column)

For Numerical variables:

Code:

```
from scipy.stats import ttest_ind
numeric=df.select_dtypes(include=np.number).columns
num_sign = pd.DataFrame()
for i in numeric:
    group1=df[df['Dry Eye Disease']=='Y'][i]
    group2=df[df['Dry Eye Disease']=='N'][i]
    t_stat, p_val = ttest_ind(group1, group2, equal_var=False)
    print(f"ttest_ind for {i}:")
    print(f"  p-value: {p_val}")
    if p_val < 0.05:
        print("  Conclusion: There is a statistically significant association between", i, "and Dry Eye Disease.")
        num_sign[i] = ['Yes']
    else:
        print("  Conclusion: There is no statistically significant association between", i, "and Dry Eye Disease.")
        num_sign[i] = ['No']
num_sign=num_sign.T # Change index 0 to column
num_sign.rename(columns = {0:'Statsitical Significant Presence'},inplace = True) # Rename that column
```

Output:

Independent Variables	Statistically Significant/ Relation with Target Column
Age	No
Sleep duration	No
Sleep quality	No
Stress level	No
Heart rate	No
Daily steps	No
Physical activity	No
Height	No
Weight	No
Average screen time	Yes
Systolic BP	No
Diastolic BP	No
Pulse_Pressure	No
BMI	No

For Categorical variables:

Code:

```
from scipy.stats import chi2_contingency
cat_sign = pd.DataFrame()
for col in categoric:
    if col !='Dry Eye Disease':
        contingency_table = pd.crosstab(df[col], df['Dry Eye Disease'])
        chi2, p, dof, expected = chi2_contingency(contingency_table)
        print(f"Chi-squared test for {col}:")
        print(f"  p-value: {p}")
        if p < 0.05:
            print("  Conclusion: There is a statistically significant association between", col, "and Dry Eye Disease.")
            cat_sign[col] = ['Yes']
        else:
            print("  Conclusion: There is no statistically significant association between", col, "and Dry Eye Disease.")
            cat_sign[col] = ['No']
cat_sign = cat_sign.T # change index 0 to column
cat_sign.rename(columns = {0:'Statistically Significance Presence'},inplace = True) # Rename the column
```

Output:

Independent Variables	Statistically Significance/ Relation with Target column
Gender	Yes
Sleep disorder	No
Wake up during night	No
Feel sleepy during day	No
Caffeine consumption	No
Alcohol consumption	No
Smoking	No
Medical issue	No
Ongoing medication	No
Smart device before bed	No
Blue-light filter	No
Discomfort Eye-strain	Yes
Redness in eye	Yes

Independent Variables	Statistically Significance/ Relation with Target column
Itchiness/Irritation in eye	Yes

f. Class imbalance and its treatment

For the target column: Since the values is having Yes class as 65% and No for 35 %. This target seems to be having no imbalance of data of data. So no balancing is done.

Pre Processing for Model

1. Transformations requirement

No Transformation is applied on any column as the distribution seems to be normal.

2. Scaling the data

Since few columns like the 'Age', 'Sleep_duration', 'Sleep_quality', 'Stress_level', 'Heart_rate','Daily_steps', 'Physical_activity','Average_screen_time', 'Systolic_BP', 'Diastolic_BP', 'Pulse_Pressure','BMI' are containing larger values and many columns are having different units so, we used Standard Scaling method in this model. (consider all these columns as numcol)

Output:

```
ss=StandardScaler()
df1[numcol]=ss.fit_transform(df1[numcol])
```

3. Encoding:

We have used dumpy encoding and map function for encoding all the categorical values

Output:

```
cols=['Sleep_disorder', 'Wake_up_during_night',
       'Feel_sleepy_during_day', 'Caffeine_consumption', 'Alcohol_consumption',
       'Smoking', 'Medical_issue', 'Ongoing_medication',
       'Smart_device_before_bed', 'Blue_light_filter', 'Discomfort_Eye_strain',
       'Redness_in_eye', 'Itchiness_Irritation_in_eye', 'Dry_Eye_Disease']
```

```
df1[cols] = df1[cols].applymap(lambda x: 1 if x == 'Y' else 0)
```

```
df1 = pd.get_dummies(df1, columns=['BP_category', 'Sleep_category', 'Screen_Time_Category'], drop_first=True,dtype=int)
```

```
df1['Gender']=df1['Gender'].apply(lambda x:1 if x=='M' else 0)
```

Modelling:

Assumptions

Check for the assumptions to be satisfied for each of the models in

1. **Assumptions for the use of SLR or Linear Regression** is the target should be a numerical column and must satisfy all these below conditions for it be Linear Regression model

Assumptions –

1. Target column should be numeric
2. There should be a linear relationship between target and independent variables
3. Must not have multicollinearity
4. Absence of Autocorrelation
5. Errors should be homoscedastic
6. Errors must follow normal distribution

But our model the target is categorical so Linear Regression is ruled out. Only Classification Models are used.

2. **Assumptions for Classification model**:

For Classification model if the target is categoric then we can apply.
Here Since the target Dry Eye Disease is categorical, we apply all the Classification model – Logistic Regression, Decision Tree, Random Forest, AdaBoost , Gradient Boost, XGBoost etc.
Also, we will perform hyper parameter tunning for the models other than Logistic Regression if any overfitting / underfitting issue is present.

Analyzing Underfit and Overfitting issues.

After running all models, we have arrived at analyzing all models' accuracy of both train and test data to find whether any of the model is performing under fit or over fit.

Train model:

	Model Name	Accuracy Score	Precision Score	Recall Score	F1 Score
0	**Base model for Logistic Regression**	**0.683357**	**0.681519**	**0.965374**	**0.798984**
0	**Decision Tree Model (Base)**	**0.624286**	**0.645548**	**0.924859**	**0.760364**
0	**Random Forest**	**1.000000**	**1.000000**	**1.000000**	**1.000000**

	Model Name	Accuracy Score	Precision Score	Recall Score	F1 Score
0	**ADA Boosting Model (Base)**	**0.680571**	**0.678457**	**0.958772**	**0.794617**
0	**GB Model base**	**0.697357**	**0.696534**	**0.939931**	**0.800132**
0	**XG model**	**0.913929**	**0.895408**	**0.981048**	**0.936274**
0	**SVM model base**	**0.707857**	**0.697021**	**0.967084**	**0.810138**
0	**LGBM Base model**	**0.729571**	**0.719839**	**0.950238**	**0.819146**

Test Model:

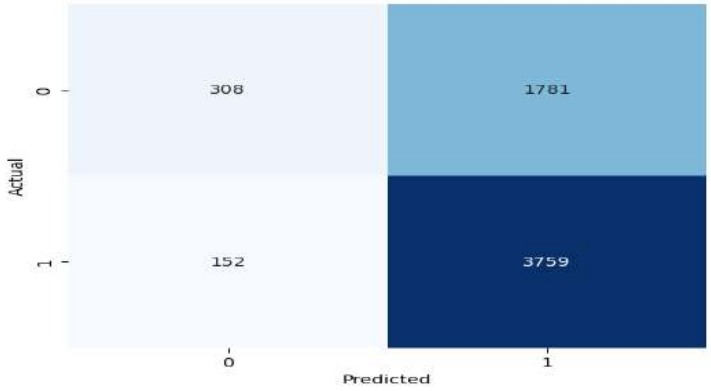
	Model Name	Accuracy Score	Precision Score	Recall Score	F1 Score
0	Base model for Logistic Regression	0.677167	0.677966	0.961391	0.795178
0	Decision Tree Model (Base)	0.568500	0.689243	0.646487	0.667181
0	Random Forest	0.700333	0.709373	0.935227	0.806791
0	AdaBoosting (Base)	0.686000	0.693355	0.951420	0.802142
0	GB Model base	0.703333	0.712035	0.934479	0.808231
0	XG model	0.654333	0.701999	0.839811	0.764746
0	SVM model	0.693500	0.699285	0.950673	0.805828
0	LGBM base model	0.698833	0.710311	0.928500	0.804881

From the above table we observe the all of the models are facing much of overfitting issues (few models face heavily but rest faced little overfitting issue.

Models:

1. **Logistic Regression**-
   Firstly, we have created a base model for train data and test data with Logistic Regression algorithm. Let's check the metrics of the model –

   Confusion Matrix:



   The model looks to perform well and predicts all values perfectly.

   Classification Report:

```
Logistic Regression
accuracy 0.6778333333333333
precision 0.6785198555956679
recall 0.9611352595244184
fi score 0.7954713786900857
classification report              precision    recall  f1-score   support

           0       0.67      0.15      0.24      2089
           1       0.68      0.96      0.80      3911

    accuracy                           0.68      6000
   macro avg       0.67      0.55      0.52      6000
weighted avg       0.68      0.68      0.60      6000
```

   Here the accuracy from model is 68%. The model predicts values perfectly as positives since recall value is higher compared to precision value for a person who is suffering from DED.

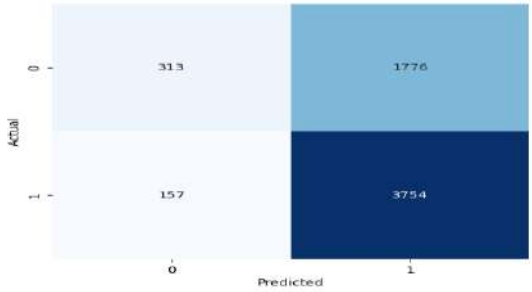
   Looks, there may be a problem of multicollinearity in the model. Checking the multicollinearity issue in model.

We have applied VIF on the X_train data and found 6 columns facing multicollinearity,

	feature	vifscore
8	Systolic_BP	3.65
9	Pulse_Pressure	3.65
0	Age	1.00
1	Sleep_duration	1.00
2	Sleep_quality	1.00
3	Stress_level	1.00
4	Heart_rate	1.00
5	Daily_steps	1.00
6	Physical_activity	1.00
7	Average_screen_time	1.00
10	BMI	1.00

Now we need not remove few columns and perform again. Checking again the metrics,

Confusion matrix:

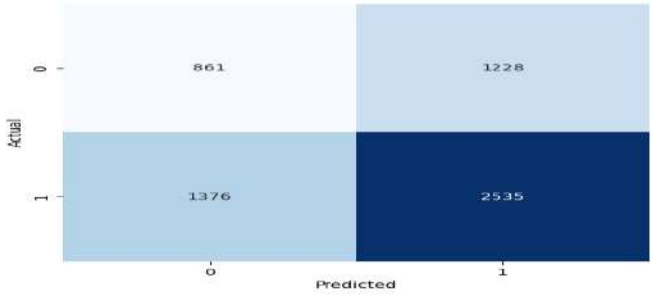


Classification Report:

```
lr_model
accuracy 0.6778333333333333
precision 0.6788426763110308
recall 0.9598568141140373
fi score 0.7952547399639869
classification report                precision    recall  f1-score   support

           0       0.67      0.15      0.24      2089
           1       0.68      0.96      0.80      3911

    accuracy                           0.68      6000
   macro avg       0.67      0.55      0.52      6000
weighted avg       0.67      0.68      0.60      6000
```

We get the same accuracy rate.

2. **Decision Tree modelling –**
   Let's, first create a base Decision Tree model using X and y data. Let's see the metrics of the model
   Confusion Matrix:



From the matrix we can observe that model is predicting the positive values more than negative class values.

Classification Report:

```
Dt_model
accuracy 0.566
precision 0.6736646292851448
recall 0.6481718230631552
fi score 0.6606724003127443
classification report                precision    recall  f1-score   support

           0       0.38      0.41      0.40      2089
           1       0.67      0.65      0.66      3911

    accuracy                           0.57      6000
   macro avg       0.53      0.53      0.53      6000
weighted avg       0.57      0.57      0.57      6000
```

Since the **accuracy** of the model is still only at **57%**. For the person having disease precision is **69%** and recall is **65%** which shows the poor model generalization or overfitting issue. So, let's use hyper parameter tunning to improve the performance of precision and recall of the model.

Tunned parameters:

```
params = {
    'criterion': ['gini', 'entropy'],
    'max_depth': [3, 5, 10, 20, None],
    'min_samples_split': [2, 5, 10],
    'min_samples_leaf': [1, 2, 4]
}
```

```
grid=GridSearchCV(estimator=dt,param_grid=params,cv=5,n_jobs=-1,verbose=1)
gs_model=grid.fit(X_train_resampled,y_train_resampled)
```

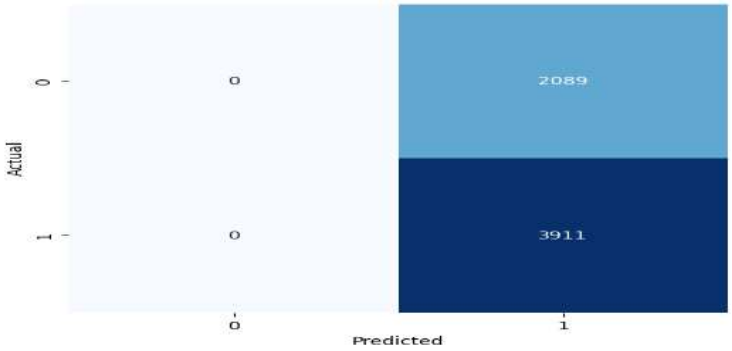
```
<IPython.core.display.Javascript object>
Fitting 5 folds for each of 90 candidates, totalling 450 fits
```

From this the above parameters the best parameters found using GridSearchCV are:

1. **Criterion:** Entropy
2. **Max_depth:** 5
3. **Min_samples_leaf:** 4
4. **Min_samples_split:** 2

After finding the best fit parameters we run the model again and check the metrics,

Confusion Matrix:



Classification Report:

```
Dt_model
accuracy 0.6518333333333334
precision 0.6518333333333334
recall 1.0
fi score 0.7892240944405207
classification report              precision    recall  f1-score   support

           0       0.00      0.00      0.00      2089
           1       0.65      1.00      0.79      3911

    accuracy                           0.65      6000
   macro avg       0.33      0.50      0.39      6000
weighted avg       0.42      0.65      0.51      6000
```

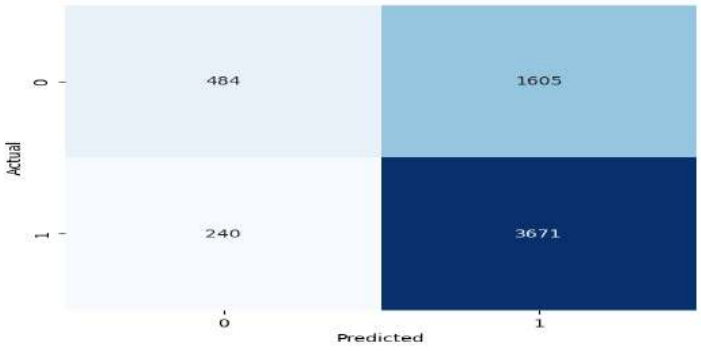
Now after tunning, accuracy of the model is improved to 65%, the precision and recall values are also improved and are predicting more of positive values to 70% and 86% for person having dry eye disease.

Important Features of the model are: Itchiness/Irritation in Eye, Redness in Eye, Discomfort in Eye Strain, Average Screen time, BMI, Pulse Pressure, Systolic BP, Physical Activity, Daily Steps, Sleep duration, Gender, Heart Rate.

3. **Random Forest modelling –**

We perform Random Forest Algorithm and create a base model for X and y data. Let's see the metrics of the Model.

Confusion Matrix:



From this we can observe that the model is predicting more true values more perfectly which is a positive sign of the model performance.

Classification Report:

```
Random Forest
accuracy 0.6925
precision 0.69579226686884
recall 0.9386346203017131
fi score 0.7991727440949167
classification report              precision    recall  f1-score   support

           0       0.67      0.23      0.34      2089
           1       0.70      0.94      0.80      3911

    accuracy                           0.69      6000
   macro avg       0.68      0.59      0.57      6000
weighted avg       0.69      0.69      0.64      6000
```

From this model Accuracy score is around 69% and for person suffering from dry eye diseases the prediction value of recall is better than precision is also higher. (indicates all positive values are predicted). Let's perform parameter tunning to improve the accuracy score.

<u>Tunning Parameters:</u>

```
params= {
    'n_estimators': [100, 200],
    'max_depth': [None, 10, 20],
    'min_samples_split': [2, 5],
    'min_samples_leaf': [1, 2],
    'max_features': ['auto', 'sqrt']
}
```

The best parameters are found by running GridSearchCV are:

N_estimators: 200

Max_depth: None

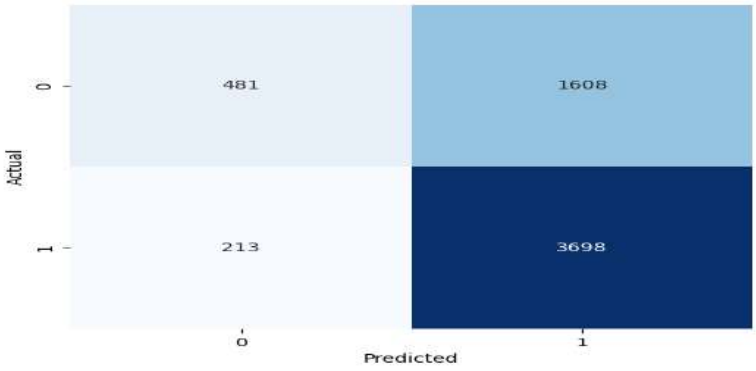
Min_samples_split: 5

Min_samples_leaf: 2

Max_features: None

Now running the model again and checking the metrics again are,

Confusion Matrix:



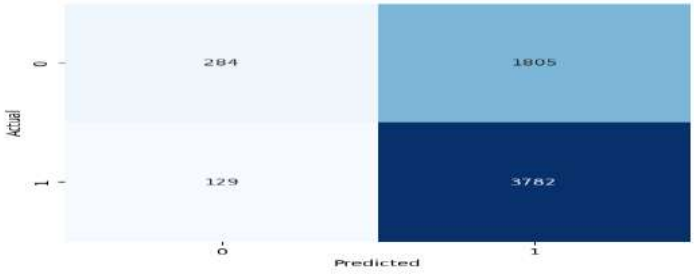
Classification Report:

```
Random Forest
accuracy 0.6965
precision 0.6969468526196758
recall 0.9455382255177703
fi score 0.8024302918520125
classification report              precision    recall  f1-score   support

           0       0.69      0.23      0.35      2089
           1       0.70      0.95      0.80      3911

    accuracy                           0.70      6000
   macro avg       0.70      0.59      0.57      6000
weighted avg       0.70      0.70      0.64      6000
```

We can now see the model accuracy is again decreased to 68 and also recall value of class 1 is remains almost same.

Important Features of the model are: All the features look important. But the top few features which shows more impact in with Target Dry Eye Disease is BMI, Physical Activity, Average Screen Time, Pulse Pressure, Sleep duration, Heart Rate, Systolic BP, Age, Daily Steps.

4. **AdaBoosting modelling –**
   Let's create AdaBoosting base model using X and y data. Let's see the metrics:
   Confusion Matrix:



Classification Report:

```
accuracy 0.6776666666666666
precision 0.6769285842133524
recall 0.9670161084121708
fi score 0.7963781848810276
classification report              precision    recall  f1-score   support

           0       0.69      0.14      0.23      2089
           1       0.68      0.97      0.80      3911

    accuracy                           0.68      6000
   macro avg       0.68      0.55      0.51      6000
weighted avg       0.68      0.68      0.60      6000
```

Accuracy score of the base model is 68 % and also recall is good than the precision value for person having dry eye disease (predicts more of positive values. So, let's try parameter tunning to improve the accuracy and other parameters.

Tunned parameters:

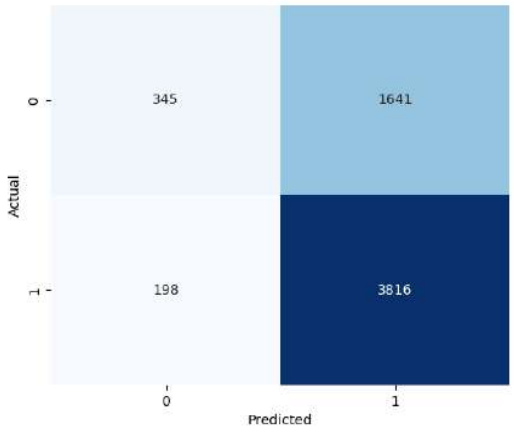
```
params= {
    'n_estimators': [50, 100, 200],
    'learning_rate': [0.01, 0.1, 0.5, 1.0]}
```

```
grid=GridSearchCV(estimator=ada,param_grid=params,cv=5,n_jobs=-1,verbose=1)
gs_model=grid.fit(X_train_resampled,y_train_resampled)
print(gs_model.best_params_)
```

```
<IPython.core.display.Javascript object>
Fitting 5 folds for each of 12 candidates, totalling 60 fits
{'learning_rate': 0.5, 'n_estimators': 200}
```

Best parameters are using GridSearchCV are:
Learning_rate: 0.5
N_estimators: 200

Once again build the model and check the metrics.
Confusion Matrix:



We can see the model is predicting more for postive values.

Classification Report:

```
tunned_model
accuracy 0.6935
precision 0.6992853216052777
recall 0.9506726457399103
fi score 0.80582831802344
classification report              precision    recall  f1-score   support

           0       0.64      0.17      0.27      1986
           1       0.70      0.95      0.81      4014

    accuracy                           0.69      6000
   macro avg       0.67      0.56      0.54      6000
weighted avg       0.68      0.69      0.63      6000
```

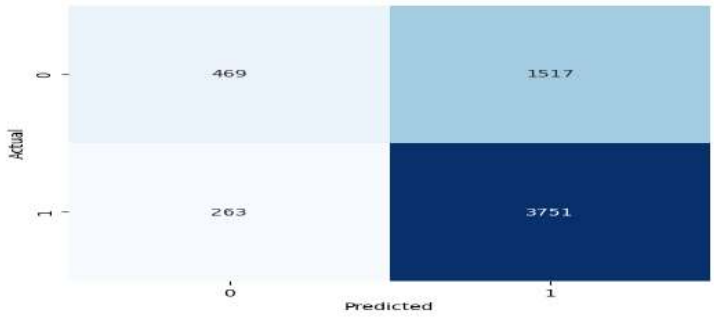
Now, the accuracy is 69%, precision is less than recall values show the model is having more positive value predictions in the model.

Important Features for this model: Discomfort Eye Strain is more having impact of Dry Eye in this model.

5. **Gradient Boosting Model-**

Creating a base model with Gradient Boosting Algorithm using X and y data. Let's check the metrics of the model.

Confusion Matrix:



The predictions looks good with more of prediction on positive values.

Classification model:

```
GB_model
accuracy 0.7033333333333334
precision 0.7120349278663629
recall 0.9344793223716991
fi score 0.808230984701573
classification report              precision    recall  f1-score   support

           0       0.64      0.24      0.35      1986
           1       0.71      0.93      0.81      4014

    accuracy                           0.70      6000
   macro avg       0.68      0.59      0.58      6000
weighted avg       0.69      0.70      0.65      6000
```

The model has accuracy of 70% with predicting for person having dry eye disease is better and good for this model. Now let's try parameter tunning for accuracy improvement.

Tunning the model:
Below are the parameters for tunning:

```
param_grid = {
    'n_estimators': [100, 200, 300],
    'learning_rate': [0.01, 0.1, 0.2],
    'max_depth': [3, 5, 7],
    'subsample': [0.8, 1.0],
    'max_features': ['sqrt', 'log2']
}
```

Best parameters are:

Learning_rate: 0.1

Max_depth: 5

Max_features: sqrt

Min_samples_split: 2

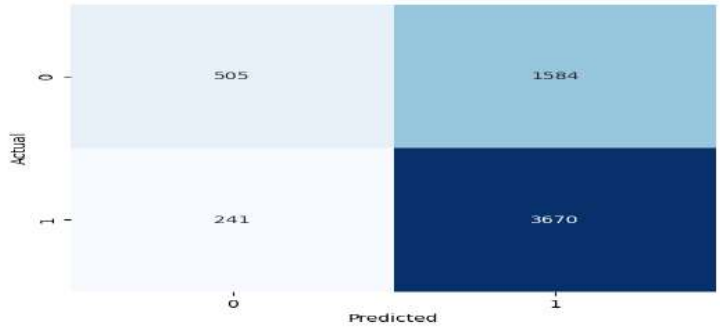
Min_samples_leaf: 3

N_estimators: 100

Subsample: 1.0

Now run the model again with these parameters and check the metrics again,

Confusion Matrix:



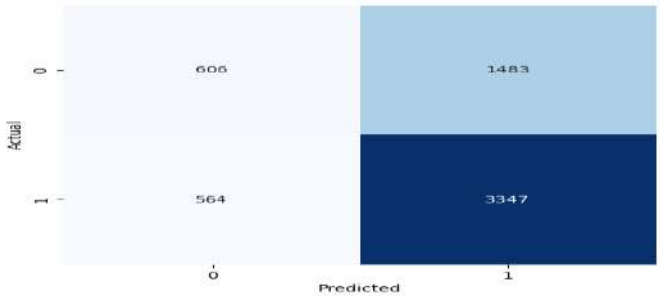
Classification Report:

```
GB-tunned
accuracy 0.6958333333333333
precision 0.698515416825276
recall 0.9383789312196369
fi score 0.800872885979269
classification report                precision    recall  f1-score   support

           0       0.68      0.24      0.36      2089
           1       0.70      0.94      0.80      3911

    accuracy                           0.70      6000
   macro avg       0.69      0.59      0.58      6000
weighted avg       0.69      0.70      0.65      6000
```

Now again the accuracy is same and the recall and precision values is also reduced.

Important Features for this model are: Almost all the Features seems to be need but in that the top most few features are Irritation/Itchiness in Eye, Redness in Eye, Discomfort in Eye Strain, BMI, Physical Activity, Pulse Pressure, Average Screen Time, Sleep duration and Heart Rate.

6. **XGBoosting Model –**
   Create a base model with XG boosting algorithm using X and y data. The metrics of the model are:
   Confusion Matrix:



Classification Report:

```
XG-model
accuracy 0.6588333333333334
precision 0.69296066252588
recall 0.8557913577090258
fi score 0.7658162681615376
classification report              precision    recall  f1-score   support

           0       0.52      0.29      0.37      2089
           1       0.69      0.86      0.77      3911

    accuracy                           0.66      6000
   macro avg       0.61      0.57      0.57      6000
weighted avg       0.63      0.66      0.63      6000
```

Accuracy score for the model is 70% and this comparative good than the random forest model. Also, the recall value is higher so let's perform some tunning of parameters.

Let's tune the model:

```
param = {
    'n_estimators': [100, 200, 500],
    'learning_rate': [0.01, 0.05, 0.1, 0.2],
    'max_depth': [3, 4, 5, 6],
    'min_child_weight': [1, 3, 5],
    'gamma': [0, 0.1, 0.5],
    'scale_pos_weight': [1, 2, 5]
}
```

Best parameter values are:
N_estimators: 100
Learning-rate: 0.01
Max_depth: 3
Min_child_weight: 1
Gamma: 0
Scale_pos_weight: 2

Let's run the model again using these parameters. The results of metrics of new model are:

Classification Report:

```
XGB Boosting
accuracy 0.6968333333333333
precision 0.699847153228888
recall 0.9365891076451035
f1 score 0.8010934937124111
classification report              precision    recall  f1-score   support

           0       0.68      0.25      0.36      2089
           1       0.70      0.94      0.80      3911

    accuracy                           0.70      6000
   macro avg       0.69      0.59      0.58      6000
weighted avg       0.69      0.70      0.65      6000
```

The Accuracy is 70% and precision is 70% for only class1 and 0 for precision 0 for class 0 which is not good predictions.

Important Features for this model are: Itchiness/Irritation in Eye, Redness in Eye Discomfort Eye Strain, Systolic BP, Daily steps, Average screen time, Alcohol Consumption, Sleep quality, Pulse Pressure, Sleep duration, Heart Rate and BMI.

7. **LightBGM Model:**
   Create a base model using X and y data for the model. The metrics are:
   Classification Report:

```
LGBM
accuracy 0.6988333333333333
precision 0.7103106537068801
recall 0.9285002491280518
fi score 0.8048806824317029
classification report              precision    recall  f1-score   support

           0       0.62      0.23      0.34      1986
           1       0.71      0.93      0.80      4014

    accuracy                           0.70      6000
   macro avg       0.66      0.58      0.57      6000
weighted avg       0.68      0.70      0.65      6000
```

The Accuracy of the model is about 70%. Precision value is greater than recall which indicates the model is predicting more to positive class values.

Let's tune the model and run build it again.

Tunned Parameters:

```
param_grid = {
    'n_estimators': [100, 200],
    'learning_rate': [0.01, 0.1],
    'max_depth': [5, 10, -1],
    'num_leaves': [31, 64],
    'subsample': [0.8, 1.0],
    'colsample_bytree': [0.8, 1.0]
}
```

Best Parameters are:

N_estimators: 200

Learning_rate: 0.01

Max_depth: 5

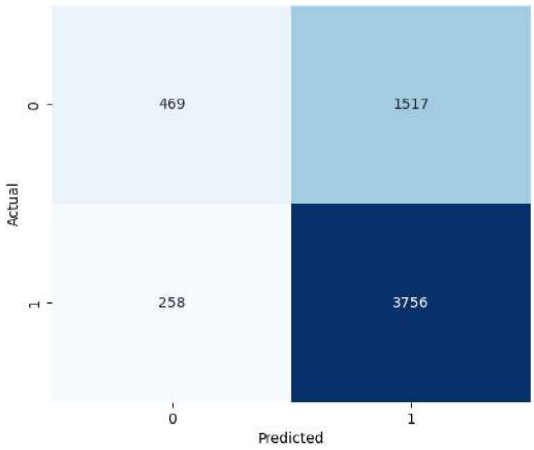
Num_leaves: 31

Subsample: 0.8

Colsample_bytress: 1

Now new model metrics are:

Confusion Matrix:



Model predicts more of Positive class values.

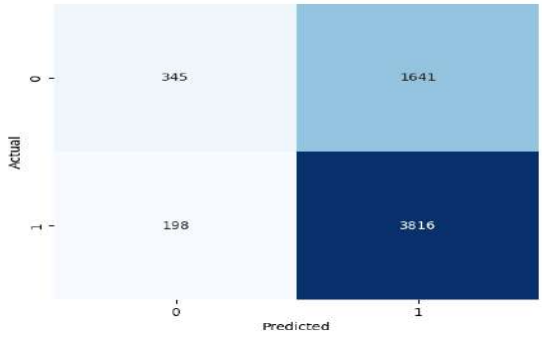
Classification Report:

```
LGBM
accuracy 0.6988333333333333
precision 0.7103106537068801
recall 0.9285002491280518
fi score 0.8048806824317029
classification report              precision    recall  f1-score   support

           0       0.62      0.23      0.34      1986
           1       0.71      0.93      0.80      4014

    accuracy                           0.70      6000
   macro avg       0.66      0.58      0.57      6000
weighted avg       0.68      0.70      0.65      6000
```

The Accuracy of the model is 70%. Also, precision value is lesser compared to recall which shows that we have models predicting on positive class values more.

Important Features are: Pulse Pressure, Average Screen Time, Physical Activity, BMI, Sleep duration, Systolic BP, Heart Rate, Age, Daily steps, Itchiness/Irritation in Eye, Redness in Eye, Discomfort in Eye Strain, Stress level and etc. have impact on target.

8. **SVM Model:**
   Create a base model using X and y data the metrices for the model are:
   Confusion Matrix:



This model is predicting more of positive values which is good sign.

Classification Report:

```
SVM_model
accuracy 0.6935
precision 0.6992853216052777
recall 0.9506726457399103
fi score 0.80582831802344
classification report              precision    recall  f1-score   support

           0       0.64      0.17      0.27      1986
           1       0.70      0.95      0.81      4014

    accuracy                           0.69      6000
   macro avg       0.67      0.56      0.54      6000
weighted avg       0.68      0.69      0.63      6000
```

The Accuracy is around 69% with precision of 70% and recall of 90% for the class 1 which tells that more of positive values is predicated**.**

Tunned Params:

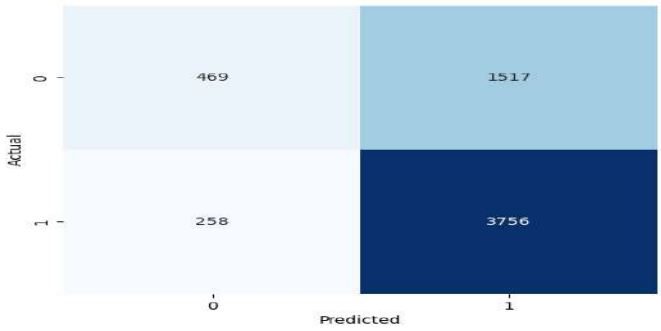
```
param_grid = {
    'C': [0.1, 1, 10],
    'kernel': ['linear', 'rbf'],
    'gamma': ['scale', 'auto']
}
```

The best tunned parameters are:

C: 1
Kernel: rbf
Gamma: scale

Let's run build the model and using these metrics now. The results are:

Confusion Matrix:



Model is predicting more of positive values.

Classification Matrix:

```
LGBM tunned
accuracy 0.7041666666666667
precision 0.7123079840697895
recall 0.9357249626307922
fi score 0.8088726176375579
classification report              precision    recall  f1-score   support

           0       0.65      0.24      0.35      1986
           1       0.71      0.94      0.81      4014

    accuracy                           0.70      6000
   macro avg       0.68      0.59      0.58      6000
weighted avg       0.69      0.70      0.66      6000
```

The Accuracy score is 70% and the precision value is lesser compared to recall value which tells the model predicts only positive values.

Important Features of model are: Pulse Pressure, Average Screen time, Physical Activity, BMI, Sleep duration, Heart Rate, Age, Daily Steps, Itchiness/Irritation in Eyes, Redness in Eye, Stress Level are having impact with target.

9. **Vote Classifier:**

The model which are used for vote classifier is :

```
voting_clf = VotingClassifier(estimators=[
    ('dt', dt_model),
    ('rf', rf),
    ('ada', ada),
    ('gb',gb),
    ('xgb', XGB),
    ('lbgm',lgbm),
], voting='soft')  # Use 'hard' for majority vote, 'soft' for probabilities
```

Close

The metrics of the model is

Confusion Matrix:

Classification Report:

```
voting
accuracy 0.7041666666666667
precision 0.7123079840697895
recall 0.9357249626307922
fi score 0.8088726176375579
classification report              precision    recall  f1-score   support

           0       0.65      0.24      0.35      1986
           1       0.71      0.94      0.81      4014

    accuracy                           0.70      6000
   macro avg       0.68      0.59      0.58      6000
weighted avg       0.69      0.70      0.66      6000
```

We can observe that the model accuracy is max to 70% and the best model used from vote for predictions is LightBGM.

Model Summary:

	Model	Accuracy	Precision	Recall	F1-Score
0	Logistic Regression	0.692333	0.697522	0.953662	0.805725
1	Decision Tree	0.704167	0.712308	0.935725	0.808873
2	Random Forest	0.700000	0.709184	0.934978	0.806576
3	Gradient Boosting	0.702500	0.711040	0.935476	0.807961
4	XGBoost	0.669000	0.669000	1.000000	0.801678
5	LightGBM	0.698833	0.710311	0.928500	0.804881
6	AdaBoost	0.704167	0.712308	0.935725	0.808873
7	SVC	0.693500	0.699285	0.950673	0.805828
8	Voting Classifier	0.704167	0.712308	0.935725	0.808873

Best model is LightBGM and Gradient Boosting Model.

Feature selection

In the data, we have applied RFE selection for different models for choosing the best features from the data for each model.

Example:

Below are the best features of Decision Tree Model

Decision Tree - 'Gender', 'Age', 'Sleep_duration', 'Sleep_quality', 'Stress_level', 'Heart_rate', 'Daily_steps', 'Physical_activity', 'Sleep_disorder','Wake_up_during_night', 'Feel_sleepy_during_day','Caffeine_consumption', 'Alcohol_consumption', 'Smoking','Medical_issue', 'Average_screen_time', 'Blue_light_filter','Discomfort_Eye_strain', 'Redness_in_eye','Itchiness_Irritation_in_eye', 'Systolic_BP', 'Diastolic_BP','Pulse_Pressure', 'BMI', 'BP_category_Hypertension Stage 1','BP_category_Hypertension Stage 2', 'BP_category_Normal', 'Sleep_category_Long', 'Sleep_category_Short','Screen_Time_Category_Low'

Code:

```
# Models
models = [dt_model,rf, ada, gb, XGB, lgbm, svm]  # your pre-trained models
model_names = ['decision tree', 'random forest', 'Ada Boost', 'GradientBoost', 'XGBoost', 'Light GBM', 'Support Vector']


# Number of features to select
n_features_to_select = 30
for model, name in zip(models, model_names):
    # RFE Feature Selection
    rfe = RFE(estimator=model, n_features_to_select=n_features_to_select)
    rfe.fit(X_train, y_train)
    selected_features = X_train.columns[rfe.support_]
    print(f'{name}\n',selected_features)
```

Conclusion:

After applying all models LightBGM Algorithm and Gradient Boosting Model are giving good predictions results and outputs. Either of the model can be used as both give almost accuracy values and optimize threshold value in same range only.

Best Features for the model:

Comparison To Benchmark

1. As a baseline, we considered a simple logistic regression model trained without any feature selection or parameter tuning. This model achieved an accuracy of approximately 68% on the test data.
2. The best-performing models—LightGBM and Gradient Boosting—achieved an accuracy of 70%, with better precision and recall values. These models significantly outperformed the baseline by leveraging advanced feature selection, hyperparameter tuning, and handling of class imbalance.
3. The final models clearly improved upon the benchmark in both predictive performance and reliability, especially in correctly identifying individuals with Dry Eye Disease (DED).

Limitations

Limitations are:

1. The dataset only includes individuals aged 18–45, limiting the model's applicability to older adults or children who may have different risk factors for DED.
2. Although addressed during modelling, the dataset shows mild class imbalance (65% Yes vs. 35% No for DED), which can affect generalization.
3. No external data was added; all new features were derived from existing columns.

External medical datasets or survey data could improve robustness.
4. Some features like stress level, sleep quality, and screen time are subjective and prone to reporting bias.
5. While addressed during modelling, multicollinearity among health features (e.g., BMI, BP) may still impact interpretability.

Future Analysis:

Applying deep learning techniques, incorporating a broader demographic, and collecting data from clinical sources would enhance model performance and reliability.

Overall, the project strengthened our understanding of end-to-end machine learning workflows and their real-world impact in the health domain.