```
from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).
```

```
# Import Dependencies -To see the graphs in the notebook.
%matplotlib inline

# Python Imports
import math,time,random,datetime

# Data Manipulation
import numpy as np
import pandas as pd

# Visualization -This is where the graphs come in.
import matplotlib.pyplot as plt
import seaborn as sns
import missingno
plt.style.use('ggplot')

## Statistical Analysis
from scipy import stats
from scipy.stats import norm,skew

# Display all Columns
pd.set_option('display.max_columns', None)

# Ignore Warnings
import warnings
warnings.filterwarnings('ignore')
```

```
# Import the train and test data.
train = pd.read_csv('/content/drive/MyDrive/genesight data/Genetic-Disorder-Prediction-main/Genetic-Disorder-Prediction-main/dataset/train.cs
test = pd.read_csv('/content/drive/MyDrive/genesight data/Genetic-Disorder-Prediction-main/Genetic-Disorder-Prediction-main/dataset/test.csv'
```

```
# Viewing the train dataset
train.head(3)
```

	Patient Id	Patient Age	Genes in mother's side	Inherited from father	Maternal gene	Paternal gene	Blood cell count (mcL)	Patient First Name	Family Name	Father's name	Mother's age	Father's age	Institute Name	Lo
0	PID0x6418	2.0	Yes	No	Yes	No	4.760603	Richard	NaN	Larre	NaN	NaN	Boston Specialty & Rehabilitation Hospital	55 02
1	PID0x25d5	4.0	Yes	Yes	No	No	4.910669	Mike	NaN	Brycen	NaN	23.0	St. Margaret's Hospital For Women	1 AV\r
2	PID0x4a82	6.0	Yes	No	No	No	4.893297	Kimberly	NaN	Nashon	41.0	22.0	NaN	

```
print("Dimensionality of the train dataset: ", train.shape)
```

```
Dimensionality of the train dataset:  (22083, 45)
```

```
train.describe()
```

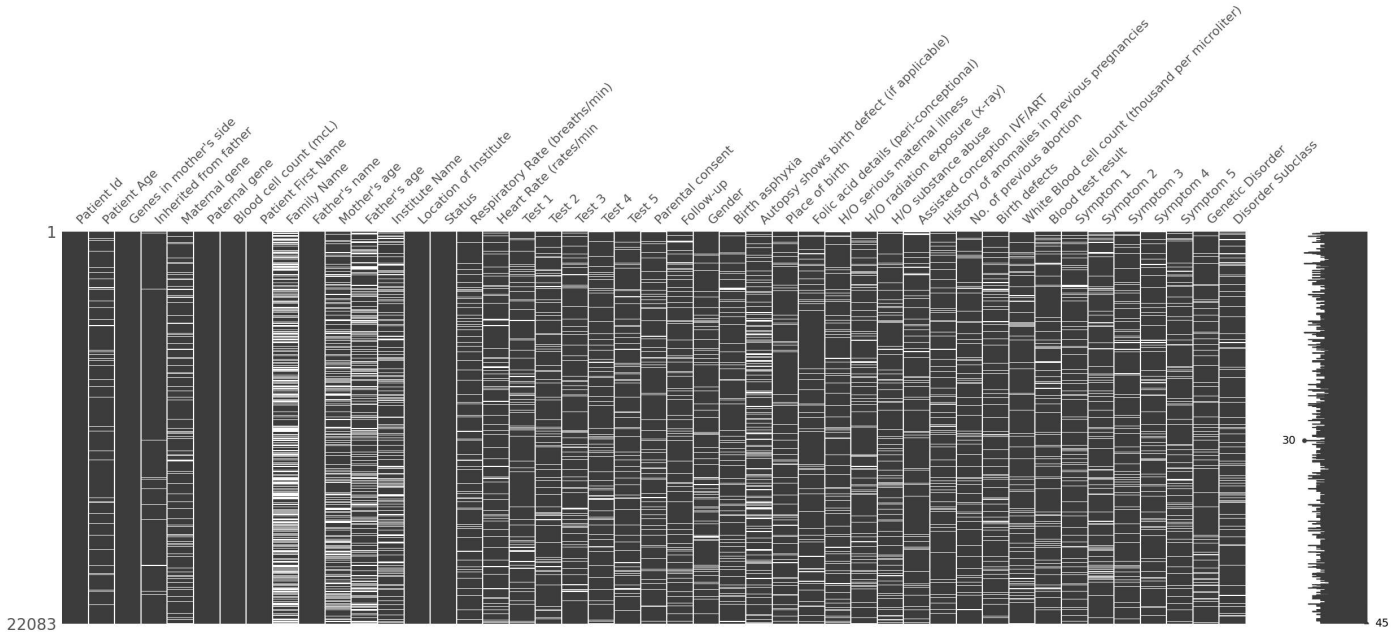
	Patient Age	Blood cell count (mcL)	Mother's age	Father's age	Test 1	Test 2	Test 3	Test 4	Test 5	No. of previous abortion	White Blood cell count (thousand per microliter)	Sym
count	20656.000000	22083.000000	16047.000000	16097.000000	19956.0	19931.0	19936.0	19943.0	19913.0	19921.000000	19935.000000	19928
mean	6.974148	4.898871	34.526454	41.972852	0.0	0.0	0.0	1.0	0.0	2.003062	7.486224	0
std	4.319475	0.199663	9.852598	13.035501	0.0	0.0	0.0	0.0	0.0	1.411919	2.653393	0
min	0.000000	4.092727	18.000000	20.000000	0.0	0.0	0.0	1.0	0.0	0.000000	3.000000	0
25%	3.000000	4.763109	26.000000	31.000000	0.0	0.0	0.0	1.0	0.0	1.000000	5.424703	0

```
## Information of the train Dataset.
train.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 22083 entries, 0 to 22082
Data columns (total 45 columns):
 #   Column                                          Non-Null Count  Dtype
---  ------                                          --------------  -----
 0   Patient Id                                      22083 non-null  object
 1   Patient Age                                     20656 non-null  float64
 2   Genes in mother's side                          22083 non-null  object
 3   Inherited from father                           21777 non-null  object
 4   Maternal gene                                   19273 non-null  object
 5   Paternal gene                                   22083 non-null  object
 6   Blood cell count (mcL)                          22083 non-null  float64
 7   Patient First Name                              22083 non-null  object
 8   Family Name                                     12392 non-null  object
 9   Father's name                                   22083 non-null  object
 10  Mother's age                                    16047 non-null  float64
 11  Father's age                                    16097 non-null  float64
 12  Institute Name                                  16977 non-null  object
 13  Location of Institute                           22083 non-null  object
 14  Status                                          22083 non-null  object
 15  Respiratory Rate (breaths/min)                  19934 non-null  object
 16  Heart Rate (rates/min                           19970 non-null  object
 17  Test 1                                          19956 non-null  float64
 18  Test 2                                          19931 non-null  float64
 19  Test 3                                          19936 non-null  float64
 20  Test 4                                          19943 non-null  float64
 21  Test 5                                          19913 non-null  float64
 22  Parental consent                                19958 non-null  object
 23  Follow-up                                       19917 non-null  object
 24  Gender                                          19910 non-null  object
 25  Birth asphyxia                                  19944 non-null  object
 26  Autopsy shows birth defect (if applicable)      17691 non-null  object
 27  Place of birth                                  19959 non-null  object
 28  Folic acid details (peri-conceptional)          19966 non-null  object
 29  H/O serious maternal illness                    19931 non-null  object
 30  H/O radiation exposure (x-ray)                  19930 non-null  object
 31  H/O substance abuse                             19888 non-null  object
 32  Assisted conception IVF/ART                     19961 non-null  object
 33  History of anomalies in previous pregnancies    19911 non-null  object
 34  No. of previous abortion                        19921 non-null  float64
 35  Birth defects                                   19929 non-null  object
 36  White Blood cell count (thousand per microliter) 19935 non-null  float64
 37  Blood test result                               19938 non-null  object
 38  Symptom 1                                       19928 non-null  float64
 39  Symptom 2                                       19861 non-null  float64
 40  Symptom 3                                       19982 non-null  float64
 41  Symptom 4                                       19970 non-null  float64
 42  Symptom 5                                       19930 non-null  float64
 43  Genetic Disorder                                19937 non-null  object
 44  Disorder Subclass                               19915 non-null  object
dtypes: float64(16), object(29)
memory usage: 7.6+ MB
```

```
# To plot a graphic of missing values
missingno.matrix(train,figsize=(30,9))
```

```
<Axes: >
```




```
# Understanding more about the missing value columns

def missing_values_table(df):
    # number of missing values
    mis_val = df.isnull().sum()

    # % of missing values
    mis_val_percent = 100 * mis_val / len(df)

    # make table # axis '0' concat along index, '1' column
    mis_val_table = pd.concat([mis_val,mis_val_percent], axis= 1)

    # rename columns
    mis_val_table_ren_columns = mis_val_table.rename(columns = {0: 'Missing Values', 1: '% of Total Values'})

    # sort by column
    mis_val_table_ren_columns = mis_val_table_ren_columns[mis_val_table_ren_columns.iloc[:,1]!= 0].sort_values(
        '% of Total Values', ascending= False).round(1) #Review
    print("Your selected dataset has " + str(df.shape[1]) + " columns and " + str(len(df)) + " observations.\n"
         "\n There are " + str(mis_val_table_ren_columns.shape[0]) + " columns that have missing values.")

    # return the dataframe with missing info
    return mis_val_table_ren_columns

missing_values_table(train)
```

```
Your selected dataset has 45 columns and 22083 observations.

There are 37 columns that have missing values.
```

	Missing Values	% of Total Values
Family Name	9691	43.9
Mother's age	6036	27.3
Father's age	5986	27.1
Institute Name	5106	23.1
Autopsy shows birth defect (if applicable)	4392	19.9
Maternal gene	2810	12.7
Symptom 2	2222	10.1
H/O substance abuse	2195	9.9
Gender	2173	9.8
History of anomalies in previous pregnancies	2172	9.8
Test 5	2170	9.8
Disorder Subclass	2168	9.8
Follow-up	2166	9.8
No. of previous abortion	2162	9.8
Symptom 1	2155	9.8
Birth defects	2154	9.8
Symptom 5	2153	9.7
H/O radiation exposure (x-ray)	2153	9.7
H/O serious maternal illness	2152	9.7
Test 2	2152	9.7
Respiratory Rate (breaths/min)	2149	9.7
White Blood cell count (thousand per microliter)	2148	9.7
Test 3	2147	9.7
Genetic Disorder	2146	9.7
Blood test result	2145	9.7
Test 4	2140	9.7
Birth asphyxia	2139	9.7
Test 1	2127	9.6
Parental consent	2125	9.6

```
# Datatypes in the dataset
train.dtypes
```

	0
Patient Id	object
Patient Age	float64
Genes in mother's side	object
Inherited from father	object
Maternal gene	object
Paternal gene	object
Blood cell count (mcL)	float64
Patient First Name	object
Family Name	object
Father's name	object
Mother's age	float64
Father's age	float64
Institute Name	object
Location of Institute	object
Status	object
Respiratory Rate (breaths/min)	object
Heart Rate (rates/min	object
Test 1	float64
Test 2	float64
Test 3	float64
Test 4	float64
Test 5	float64
Parental consent	object
Follow-up	object
Gender	object
Birth asphyxia	object
Autopsy shows birth defect (if applicable)	object
Place of birth	object
Folic acid details (peri-conceptional)	object
H/O serious maternal illness	object
H/O radiation exposure (x-ray)	object

```
train.head(3)
```

	Patient Id	Patient Age	Genes in mother's side	Inherited from father	Maternal gene	Paternal gene	Blood cell count (mcL)	Patient First Name	Family Name	Father's name	Mother's age	Father's age	Institute Name	Lo
0	PID0x6418	2.0	Yes	No	Yes	No	4.760603	Richard	NaN	Larre	NaN	NaN	Boston Specialty & Rehabilitation Hospital	55 02
1	PID0x25d5	4.0	Yes	Yes	No	No	4.910669	Mike	NaN	Brycen	NaN	23.0	St. Margaret's Hospital For Women	1 AV\r
2	PID0x4a82	6.0	Yes	No	No	No	4.893297	Kimberly	NaN	Nashon	41.0	22.0	NaN	

Disorder Subclass	object

```
## Checking For Duplicates
train['Patient Id'].duplicated().any()
```

```
False
```

```
## Basic Statistics
train['Patient Age'].describe()
```

	Patient Age
count	20656.000000
mean	6.974148
std	4.319475
min	0.000000
25%	3.000000
50%	7.000000
75%	11.000000
max	14.000000

```
## Missing Values
train['Patient Age'].isnull().any()
```

```
True
```

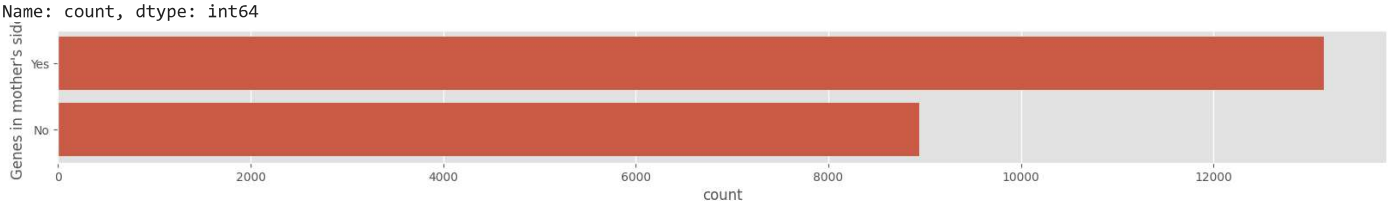
```
## Unique Categories
train["Genes in mother's side"].unique()
```

```
array(['Yes', 'No'], dtype=object)
```

```
# Is there any Gene Defect in the Mother?
fig = plt.figure(figsize=(20,2))
sns.countplot(y= "Genes in mother's side",data= train)
#sns.countplot(x= "Genes in mother's side",data= train)             --> You can use this to get a vertical graph.

print(train["Genes in mother's side"].value_counts())             ##This will help us see the exact number along with the graph.
```

```
Genes in mother's side
Yes    13143
No      8940
Name: count, dtype: int64
```



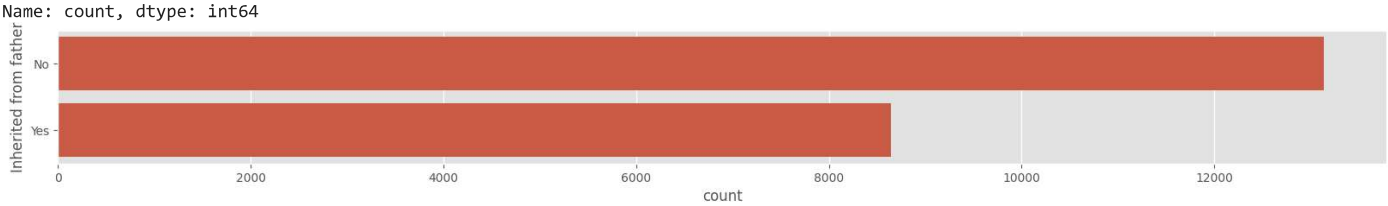

```
## Unique Categories
train["Inherited from father"].unique()
```

```
array(['No', 'Yes', nan], dtype=object)
```

```
# Is there any Gene Defect in the Father?
fig = plt.figure(figsize=(20,2))
sns.countplot(y= "Inherited from father",data= train)

print(train["Inherited from father"].value_counts())
```

```
Inherited from father
No     13133
Yes     8644
Name: count, dtype: int64
```



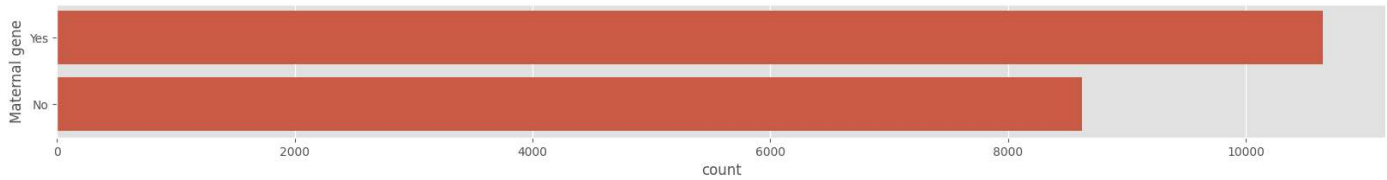

```
## Unique Categories
train["Maternal gene"].unique()
```

```
array(['Yes', 'No', nan], dtype=object)
```

```
# Is there any Gene Defect in the Mothers Side?
fig = plt.figure(figsize=(20,2))
sns.countplot(y= "Maternal gene",data= train)

print(train["Maternal gene"].value_counts())
```

```
Maternal gene
Yes    10647
No      8626
Name: count, dtype: int64
```



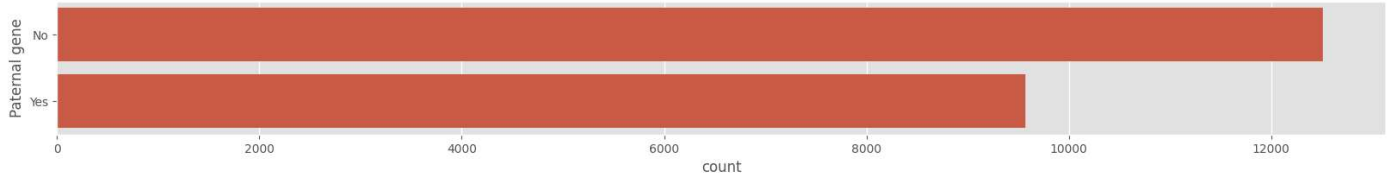

```
## Unique Categories
train["Paternal gene"].unique()
```

```
array(['No', 'Yes'], dtype=object)
```

```
# Is there any Gene Defect in the Fathers Side?
fig = plt.figure(figsize=(20,2))
sns.countplot(y= "Paternal gene",data= train)

print(train["Paternal gene"].value_counts())
```

```
Paternal gene
No     12508
Yes     9575
Name: count, dtype: int64
```




```
## Basic Statistics
train['Blood cell count (mcL)'].describe()
```

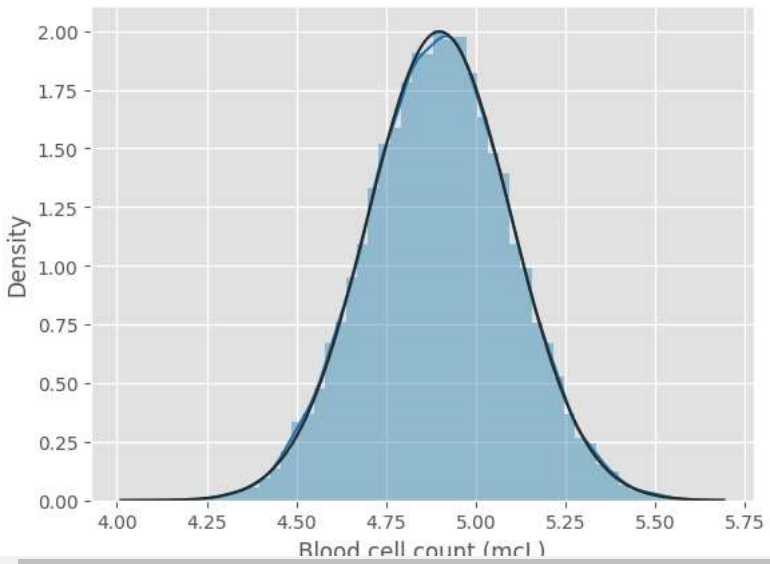
	Blood cell count (mcL)
count	22083.000000
mean	4.898871
std	0.199663
min	4.092727
25%	4.763109
50%	4.899399
75%	5.033830
max	5.609829

```
## Missing Values
train['Blood cell count (mcL)'].isnull().any()
```

```
False
```

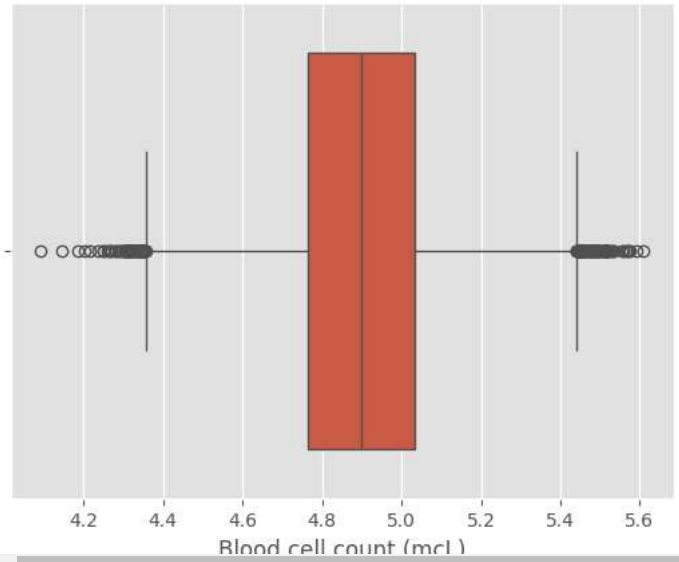
```
## Distribution of the Feature
sns.distplot(train['Blood cell count (mcL)'], fit= norm, color= 'tab:blue')
```

```
<Axes: xlabel='Blood cell count (mcL)', ylabel='Density'>
```




```
## Checking For Outliers
sns.boxplot(x= train['Blood cell count (mcL)'])
```

```
<Axes: xlabel='Blood cell count (mcL)'>
```




```
## Missing Values
train["Mother's age"].isnull().any()
```

```
True
```

```
## Basic Statistics
train["Mother's age"].describe()
```

	Mother's age
count	16047.000000
mean	34.526454
std	9.852598
min	18.000000
25%	26.000000
50%	35.000000
75%	43.000000
max	51.000000

```
## Missing Values
train["Father's age"].isnull().any()
```

```
True
```

```
## Basic Statistics
train["Father's age"].describe()
```

	Father's age
count	16097.000000
mean	41.972852
std	13.035501
min	20.000000
25%	31.000000
50%	42.000000
75%	53.000000
max	64.000000

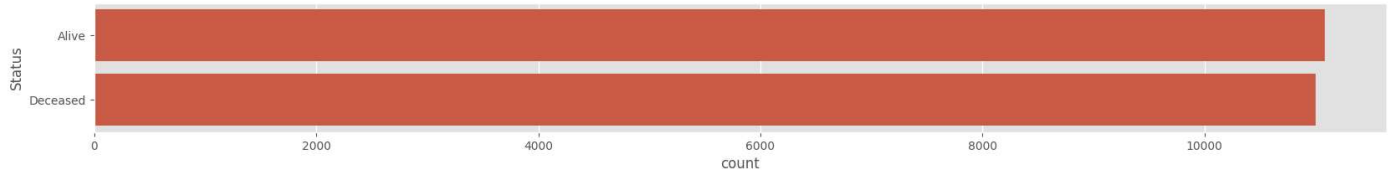
```
## Unique Categories
train["Status"].unique()
```

```
array(['Alive', 'Deceased'], dtype=object)
```

```
# Understanding the Status of Patients
fig = plt.figure(figsize=(20,2))
sns.countplot(y= "Status",data= train)

print(train["Status"].value_counts())
```

```
Status
Alive       11083
Deceased    11000
Name: count, dtype: int64
```



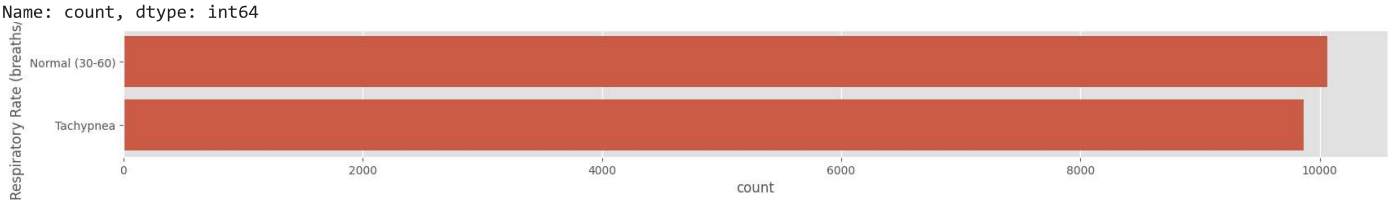

```
## Unique Categories
train["Respiratory Rate (breaths/min)"].unique()
```

```
array(['Normal (30-60)', 'Tachypnea', nan], dtype=object)
```

```
# Understanding the Respiratory Rate of Patients
fig = plt.figure(figsize=(20,2))
sns.countplot(y= "Respiratory Rate (breaths/min)",data= train)

print(train["Respiratory Rate (breaths/min)"].value_counts())
```

```
Respiratory Rate (breaths/min)
Normal (30-60)    10065
Tachypnea          9869
Name: count, dtype: int64
```



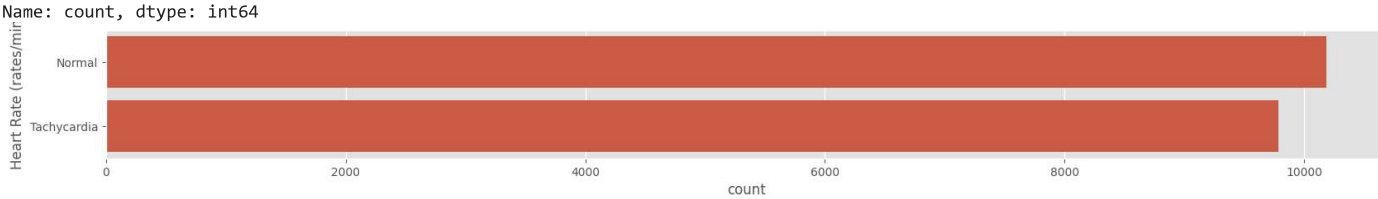

```
## Unique Categories
train["Heart Rate (rates/min"].unique()
```

```
array(['Normal', 'Tachycardia', nan], dtype=object)
```

```
# Understanding the Heart Rate of Patients
fig = plt.figure(figsize=(20,2))
sns.countplot(y= "Heart Rate (rates/min",data= train)
```

```
print(train["Heart Rate (rates/min"].value_counts())
```

```
Heart Rate (rates/min
Normal         10187
Tachycardia     9783
Name: count, dtype: int64
```




```
print(f"Uniqueness for Test 1 is: {train['Test 1'].unique()}")
print()
print(f"Uniqueness for Test 2 is: {train['Test 2'].unique()}")
print()
print(f"Uniqueness for Test 3 is: {train['Test 3'].unique()}")
print()
print(f"Uniqueness for Test 4 is: {train['Test 4'].unique()}")
print()
print(f"Uniqueness for Test 5 is: {train['Test 5'].unique()}")
```

```
Uniqueness for Test 1 is: [ 0. nan]

Uniqueness for Test 2 is: [nan  0.]

Uniqueness for Test 3 is: [nan  0.]

Uniqueness for Test 4 is: [ 1. nan]

Uniqueness for Test 5 is: [ 0. nan]
```

```
print(f"Count for Test 1 is: {train['Test 1'].value_counts()}")
print()
print(f"Count for Test 2 is: {train['Test 2'].value_counts()}")
print()
print(f"Count for Test 3 is: {train['Test 3'].value_counts()}")
print()
print(f"Count for Test 4 is: {train['Test 4'].value_counts()}")
print()
print(f"Count for Test 5 is: {train['Test 5'].value_counts()}")
```

```
Count for Test 1 is: Test 1
0.0    19956
Name: count, dtype: int64

Count for Test 2 is: Test 2
0.0    19931
Name: count, dtype: int64

Count for Test 3 is: Test 3
0.0    19936
Name: count, dtype: int64

Count for Test 4 is: Test 4
1.0    19943
Name: count, dtype: int64

Count for Test 5 is: Test 5
0.0    19913
Name: count, dtype: int64
```

```
## Unique Categories
train["Parental consent"].unique()
```

```
array(['Yes', nan], dtype=object)
```

```
## Count of Categories
train["Parental consent"].value_counts()
```

Parental consent	count
Yes	19958

```
## Unique Categories
train["Follow-up"].unique()
```

```
array(['High', 'Low', nan], dtype=object)
```

```
# Understanding the Risk Cases
fig = plt.figure(figsize=(20,2))
sns.countplot(y= "Follow-up",data= train)

print(train["Follow-up"].value_counts())
```

```
Follow-up
Low     10040
High     9877
Name: count, dtype: int64
```

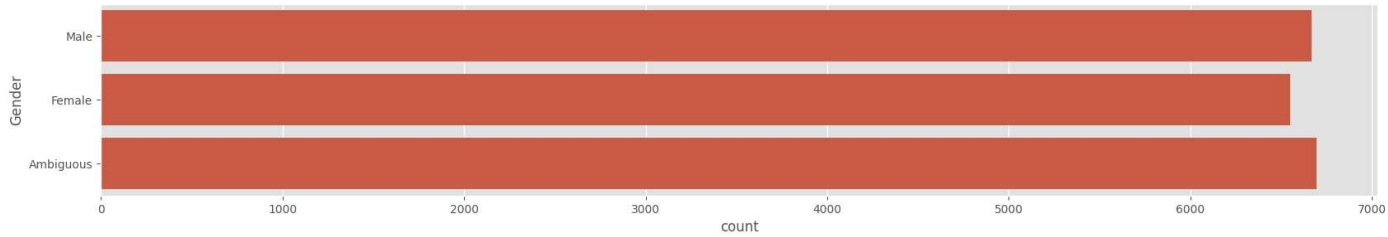

```
## Unique Categories
train["Gender"].unique()
```

```
array([nan, 'Male', 'Female', 'Ambiguous'], dtype=object)
```

```
# Understanding the Gender distribution
fig = plt.figure(figsize=(20,3))
sns.countplot(y= "Gender",data= train)

print(train["Gender"].value_counts())
```

```
Gender
Ambiguous    6695
Male         6666
Female       6549
Name: count, dtype: int64
```



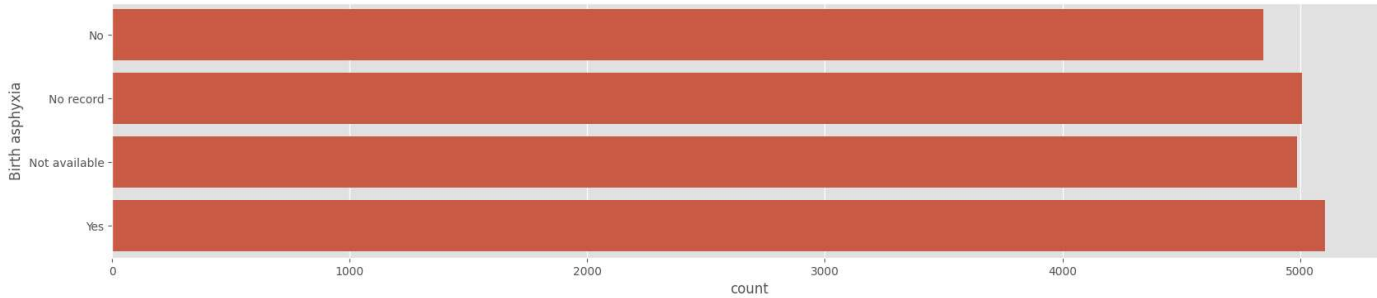

```
## Unique Categories
train["Birth asphyxia"].unique()
```

```
array([nan, 'No', 'No record', 'Not available', 'Yes'], dtype=object)
```

```
# Understanding Birth asphyxia distribution
fig = plt.figure(figsize=(20,4))
sns.countplot(y= "Birth asphyxia",data= train)

print(train["Birth asphyxia"].value_counts())
```

```
Birth asphyxia
Yes              5106
No record        5008
Not available    4986
No               4844
Name: count, dtype: int64
```



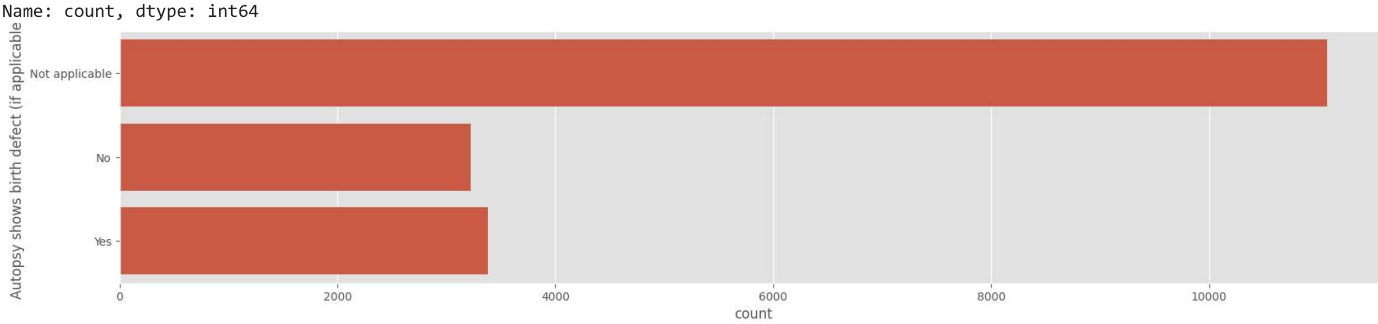

```
## Unique Categories
train["Autopsy shows birth defect (if applicable)"].unique()
```

```
array(['Not applicable', nan, 'No', 'Yes'], dtype=object)
```

```
# Understanding Birth defect distribution
fig = plt.figure(figsize=(20,4))
sns.countplot(y= "Autopsy shows birth defect (if applicable)",data= train)

print(train["Autopsy shows birth defect (if applicable)"].value_counts())
```

```
Autopsy shows birth defect (if applicable)
Not applicable    11083
Yes                3383
No                 3225
Name: count, dtype: int64
```




```
## Checking the Status of Patients with Missing Autopsy Reports
X = train[train["Autopsy shows birth defect (if applicable)"].isnull() == True]
X['Status'].value_counts()
```

Status	count
Deceased	4392

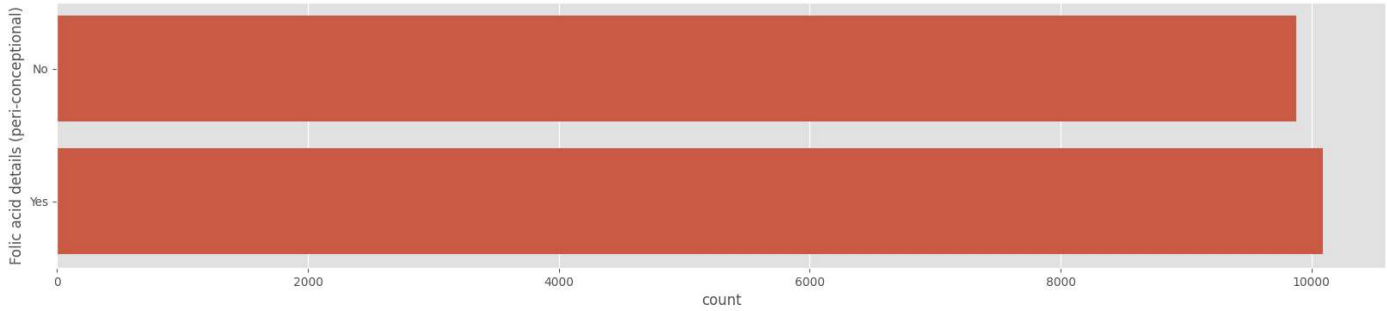
```
## Unique Categories
train["Folic acid details (peri-conceptional)"].unique()
```

```
array(['No', 'Yes', nan], dtype=object)
```

```
# Understanding Folic acid details
fig = plt.figure(figsize=(20,4))
sns.countplot(y= "Folic acid details (peri-conceptional)",data= train)

print(train["Folic acid details (peri-conceptional)"].value_counts())
```

```
Folic acid details (peri-conceptional)
Yes    10087
No      9879
Name: count, dtype: int64
```



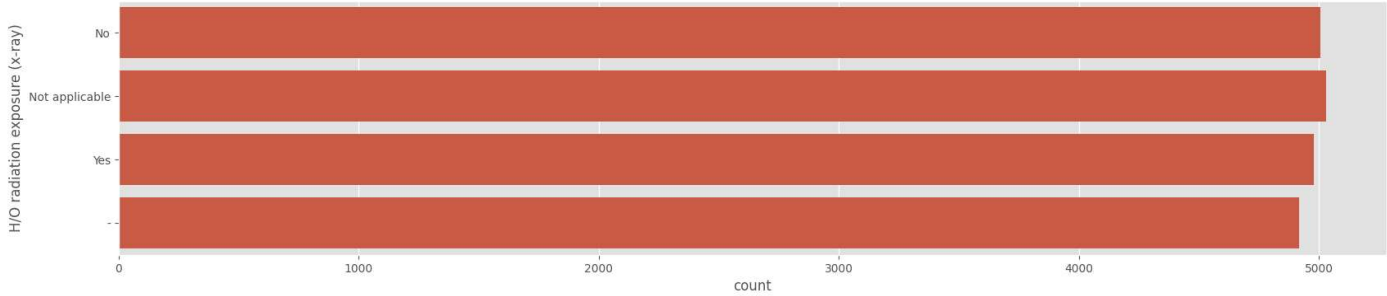

```
## Unique Categories
train["H/O radiation exposure (x-ray)"].unique()
```

```
array(['No', 'Not applicable', 'Yes', '-', nan], dtype=object)
```

```
# Understanding Radiation Exposure details
fig = plt.figure(figsize=(20,4))
sns.countplot(y= "H/O radiation exposure (x-ray)",data= train)

print(train["H/O radiation exposure (x-ray)"].value_counts())
```

```
H/O radiation exposure (x-ray)
Not applicable    5029
No                5005
Yes               4980
-                 4916
Name: count, dtype: int64
```



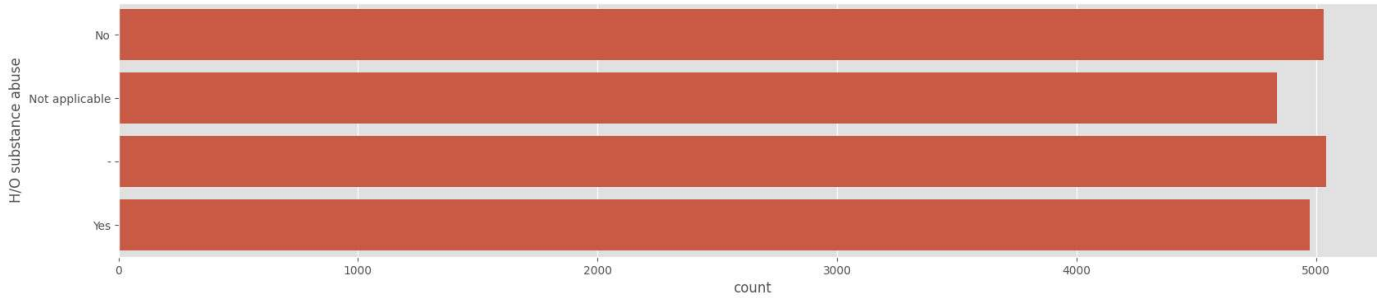

```
## Unique Categories
train["H/O substance abuse"].unique()
```

```
array(['No', 'Not applicable', nan, '-', 'Yes'], dtype=object)
```

```
# Understanding Substance Abuse details
fig = plt.figure(figsize=(20,4))
sns.countplot(y= "H/O substance abuse",data= train)

print(train["H/O substance abuse"].value_counts())
```

```
H/O substance abuse
-                 5042
No                5033
Yes               4975
Not applicable    4838
Name: count, dtype: int64
```



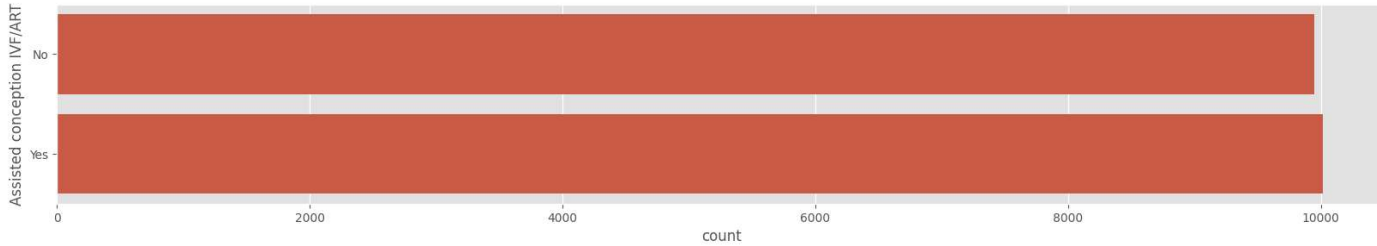

```
## Unique Categories
train["Assisted conception IVF/ART"].unique()
```

```
array(['No', 'Yes', nan], dtype=object)
```

```
# Understanding Assisted Conception details
fig = plt.figure(figsize=(20,3))
sns.countplot(y= "Assisted conception IVF/ART",data= train)

print(train["Assisted conception IVF/ART"].value_counts())
```

```
Assisted conception IVF/ART
Yes    10012
No      9949
Name: count, dtype: int64
```



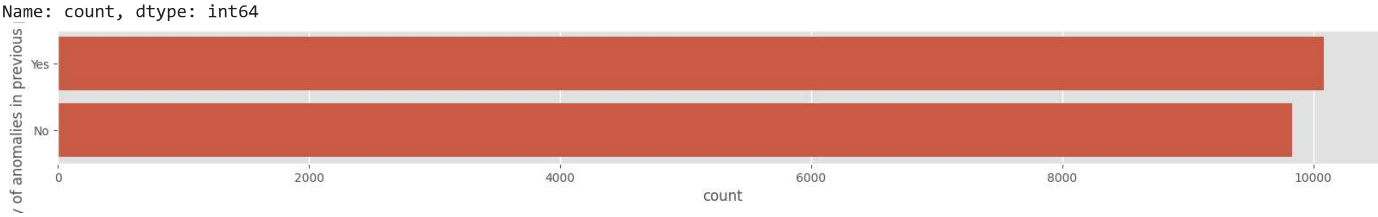

```
## Unique Categories
train["History of anomalies in previous pregnancies"].unique()
```

```
array(['Yes', 'No', nan], dtype=object)
```

```
# Understanding Anomalies in Previous Pregnancies details
fig = plt.figure(figsize=(20,2))
sns.countplot(y= "History of anomalies in previous pregnancies",data= train)

print(train["History of anomalies in previous pregnancies"].value_counts())
```

```
History of anomalies in previous pregnancies
Yes    10082
No      9829
Name: count, dtype: int64
```



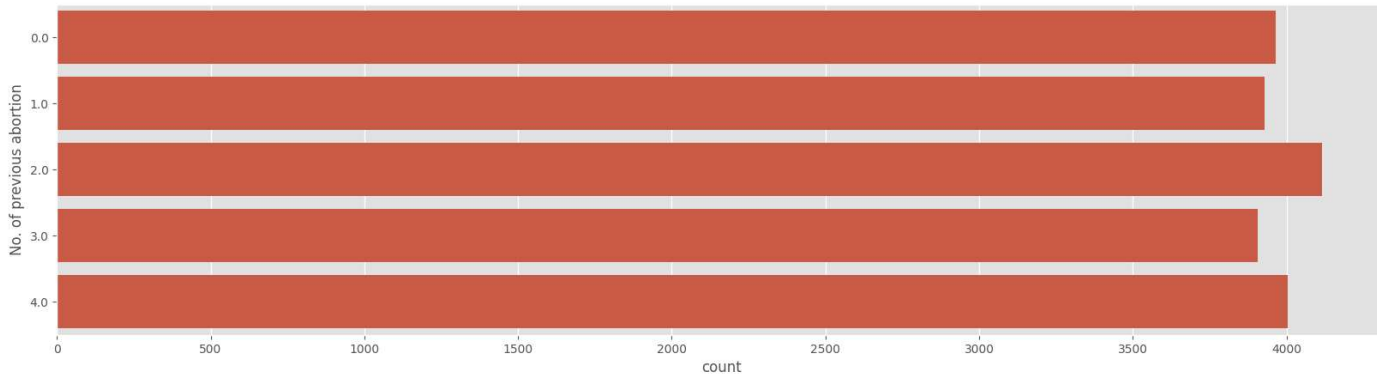

```
## Unique Categories
train["No. of previous abortion"].unique()
```

```
array([nan,  4.,  1.,  0.,  3.,  2.])
```

```
# Understanding Abortion details
fig = plt.figure(figsize=(20,5))
sns.countplot(y= "No. of previous abortion",data= train)

print(train["No. of previous abortion"].value_counts())
```

```
No. of previous abortion
2.0    4117
4.0    4005
0.0    3964
1.0    3928
3.0    3907
Name: count, dtype: int64
```




```
## Unique Categories
train["Birth defects"].unique()
```

```
array([nan, 'Multiple', 'Singular'], dtype=object)
```

```
# Understanding Birth Defects details
fig = plt.figure(figsize=(20,2))
sns.countplot(y= "Birth defects",data= train)

print(train["Birth defects"].value_counts())
```

```
Birth defects
Singular    9977
Multiple    9952
Name: count, dtype: int64
```


```
## Missing Values
train["White Blood cell count (thousand per microliter)"].isnull().any()
```

```
True
```

```
## Basic Statistics
train['White Blood cell count (thousand per microliter)'].describe()
```

	White Blood cell count (thousand per microliter)
count	19935.000000
mean	7.486224
std	2.653393

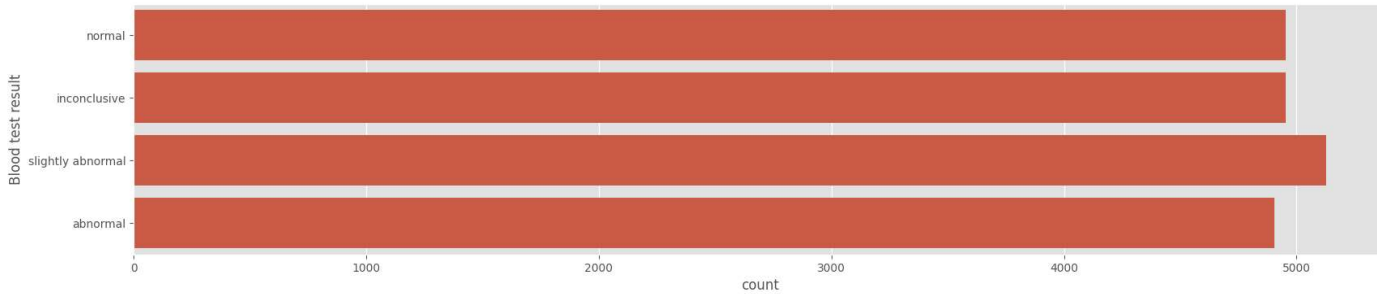
```
## Unique Categories
train["Blood test result"].unique()
```

```
array([nan, 'normal', 'inconclusive', 'slightly abnormal', 'abnormal'],
      dtype=object)
```

```
# Understanding Blood Test Result details
fig = plt.figure(figsize=(20,4))
sns.countplot(y= "Blood test result",data= train)

print(train["Blood test result"].value_counts())
```

```
Blood test result
slightly abnormal    5128
normal               4954
inconclusive         4952
abnormal             4904
Name: count, dtype: int64
```




```
print(f"Uniqueness for Symptom 1 is: {train['Symptom 1'].unique()}")
print()
print(f"Uniqueness for Symptom 2 is: {train['Symptom 2'].unique()}")
print()
print(f"Uniqueness for Symptom 3 is: {train['Symptom 3'].unique()}")
print()
print(f"Uniqueness for Symptom 4 is: {train['Symptom 4'].unique()}")
print()
print(f"Uniqueness for Symptom 5 is: {train['Symptom 5'].unique()}")
```

```
Uniqueness for Symptom 1 is: [ 1.  0. nan]

Uniqueness for Symptom 2 is: [ 1. nan  0.]

Uniqueness for Symptom 3 is: [ 1.  0. nan]

Uniqueness for Symptom 4 is: [ 1.  0. nan]

Uniqueness for Symptom 5 is: [ 1.  0. nan]
```

```
print(f"Count for Symptom 1 is: {train['Symptom 1'].value_counts()}")
print()
print(f"Count for Symptom 2 is: {train['Symptom 2'].value_counts()}")
```