Import data and package stuff

```
!pip install numpy pandas matplotlib seaborn optuna plotly scikit-learn imbalanced-learn catboost

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import optuna, warnings, random
import plotly.express as px
import plotly.io as pio
import plotly.graph_objects as go
from sklearn.metrics import confusion_matrix, classification_report,accuracy_score
from sklearn.preprocessing import MinMaxScaler
from imblearn.over_sampling import SMOTE
from sklearn.model_selection import train_test_split
from catboost import CatBoostClassifier

plt.style.use('dark_background')
warnings.simplefilter('ignore', category=FutureWarning)
```

```
Requirement already satisfied: numpy in /usr/local/lib/python3.10/dist-packages (1.25.2)
Requirement already satisfied: pandas in /usr/local/lib/python3.10/dist-packages (2.0.3)
Requirement already satisfied: matplotlib in /usr/local/lib/python3.10/dist-packages (3.7.1)
Requirement already satisfied: seaborn in /usr/local/lib/python3.10/dist-packages (0.13.1)
Requirement already satisfied: optuna in /usr/local/lib/python3.10/dist-packages (3.6.1)
Requirement already satisfied: plotly in /usr/local/lib/python3.10/dist-packages (5.15.0)
Requirement already satisfied: scikit-learn in /usr/local/lib/python3.10/dist-packages (1.2.2)
Requirement already satisfied: imbalanced-learn in /usr/local/lib/python3.10/dist-packages (0.10.1)
Requirement already satisfied: catboost in /usr/local/lib/python3.10/dist-packages (1.2.5)
Requirement already satisfied: python-dateutil>=2.8.2 in /usr/local/lib/python3.10/dist-packages (from pandas) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-packages (from pandas) (2023.4)
Requirement already satisfied: tzdata>=2022.1 in /usr/local/lib/python3.10/dist-packages (from pandas) (2024.1)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (1.2.1)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (4.51.0)
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (1.4.5)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (24.0)
Requirement already satisfied: pillow>=6.2.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (9.4.0)
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (3.1.2)
Requirement already satisfied: alembic>=1.5.0 in /usr/local/lib/python3.10/dist-packages (from optuna) (1.13.1)
Requirement already satisfied: colorlog in /usr/local/lib/python3.10/dist-packages (from optuna) (6.8.2)
Requirement already satisfied: sqlalchemy>=1.3.0 in /usr/local/lib/python3.10/dist-packages (from optuna) (2.0.29)
Requirement already satisfied: tqdm in /usr/local/lib/python3.10/dist-packages (from optuna) (4.66.4)
Requirement already satisfied: PyYAML in /usr/local/lib/python3.10/dist-packages (from optuna) (6.0.1)
Requirement already satisfied: tenacity>=6.2.0 in /usr/local/lib/python3.10/dist-packages (from plotly) (8.2.3)
Requirement already satisfied: scipy>=1.3.2 in /usr/local/lib/python3.10/dist-packages (from scikit-learn) (1.11.4)
Requirement already satisfied: joblib>=1.1.1 in /usr/local/lib/python3.10/dist-packages (from scikit-learn) (1.4.2)
Requirement already satisfied: threadpoolctl>=2.0.0 in /usr/local/lib/python3.10/dist-packages (from scikit-learn) (3.5.0)
Requirement already satisfied: graphviz in /usr/local/lib/python3.10/dist-packages (from catboost) (0.20.3)
Requirement already satisfied: six in /usr/local/lib/python3.10/dist-packages (from catboost) (1.16.0)
Requirement already satisfied: Mako in /usr/local/lib/python3.10/dist-packages (from alembic>=1.5.0->optuna) (1.3.3)
Requirement already satisfied: typing-extensions>=4 in /usr/local/lib/python3.10/dist-packages (from alembic>=1.5.0->optuna) (4.11.0)
Requirement already satisfied: greenlet!=0.4.17 in /usr/local/lib/python3.10/dist-packages (from sqlalchemy>=1.3.0->optuna) (3.0.3)
Requirement already satisfied: MarkupSafe>=0.9.2 in /usr/local/lib/python3.10/dist-packages (from Mako->alembic>=1.5.0->optuna) (2.1.5)
```

```
ds = pd.read_csv('AIDS_Classification.csv')

ds
```

	time	trt	age	wtkg	hemo	homo	drugs	karnof	oprior	z30	...	str2	strat
0	948	2	48	89.8128	0	0	0	100	0	0	...	0	1
1	1002	3	61	49.4424	0	0	0	90	0	1	...	1	3
2	961	3	45	88.4520	0	1	1	90	0	1	...	1	3
3	1166	3	47	85.2768	0	1	0	100	0	1	...	1	3
4	1090	0	43	66.6792	0	1	0	100	0	1	...	1	3
...	...	...	...	...	...	...	...	...	...	...	...	...	...
2134	1091	3	21	53.2980	1	0	0	100	0	1	...	1	3
2135	395	0	17	102.9672	1	0	0	100	0	1	...	1	3
2136	1104	2	53	69.8544	1	1	0	90	0	1	...	1	3
2137	465	0	14	60.0000	1	0	0	100	0	0	...	0	1
2138	1045	3	45	77.3000	1	0	0	100	0	0	...	0	1

2139 rows × 23 columns

Data Insights

```
pd.DataFrame(ds.isna().sum()).T.style.background_gradient(cmap='rainbow')
```

	time	trt	age	wtkg	hemo	homo	drugs	karnof	oprior	z30	preanti	race	gender	s
0	0	0	0	0	0	0	0	0	0	0	0	0	0	

```
ds.isna().sum()
```

```
time        0
trt         0
age         0
wtkg        0
hemo        0
homo        0
drugs       0
karnof      0
oprior      0
z30         0
preanti     0
race        0
gender      0
str2        0
strat       0
symptom     0
treat       0
offtrt      0
cd40        0
cd420       0
cd80        0
cd820       0
infected    0
dtype: int64
```

```
ds.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2139 entries, 0 to 2138
Data columns (total 23 columns):
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   time      2139 non-null   int64  
 1   trt       2139 non-null   int64  
 2   age       2139 non-null   int64  
 3   wtkg      2139 non-null   float64
 4   hemo      2139 non-null   int64  
 5   homo      2139 non-null   int64  
 6   drugs     2139 non-null   int64  
 7   karnof    2139 non-null   int64  
 8   oprior    2139 non-null   int64  
 9   z30       2139 non-null   int64  
 10  preanti   2139 non-null   int64  
 11  race      2139 non-null   int64
 12  gender    2139 non-null   int64
 13  str2      2139 non-null   int64
 14  strat     2139 non-null   int64
 15  symptom   2139 non-null   int64
 16  treat     2139 non-null   int64
 17  offtrt    2139 non-null   int64
 18  cd40      2139 non-null   int64
 19  cd420     2139 non-null   int64
 20  cd80      2139 non-null   int64
 21  cd820     2139 non-null   int64
 22  infected  2139 non-null   int64
dtypes: float64(1), int64(22)
memory usage: 384.5 KB
```

```
ds.describe()
```

	time	trt	age	wtkg	hemo	homo	
count	2139.000000	2139.000000	2139.000000	2139.000000	2139.000000	2139.000000	2139.0
mean	879.098177	1.520804	35.248247	75.125311	0.084151	0.661057	0.1
std	292.274324	1.127890	8.709026	13.263164	0.277680	0.473461	0.3
min	14.000000	0.000000	12.000000	31.000000	0.000000	0.000000	0.0
25%	727.000000	1.000000	29.000000	66.679200	0.000000	0.000000	0.0
50%	997.000000	2.000000	34.000000	74.390400	0.000000	1.000000	0.0
75%	1091.000000	3.000000	40.000000	82.555200	0.000000	1.000000	0.0
max	1231.000000	3.000000	70.000000	159.939360	1.000000	1.000000	1.0

8 rows × 23 columns

```python
def to_categorical(df):
    cat_columns = [
        'trt',
        'hemo',
        'homo',
        'drugs',
        'oprior',
        'z30',
        'race',
        'gender',
        'str2',
        'strat',
        'symptom',
        'treat',
        'offtrt'
    ]
    for i in cat_columns:
        df[i] = pd.Categorical(df[i])
    return df
```

```python
copy_ds=ds.copy()
to_categorical(copy_ds).dtypes
```

```
time          int64
trt        category
age           int64
wtkg        float64
hemo       category
homo       category
drugs      category
karnof        int64
oprior     category
z30        category
preanti       int64
race       category
gender     category
str2       category
strat      category
symptom    category
treat      category
offtrt     category
cd40          int64
```

```
cd420          int64
cd80           int64
cd820          int64
infected       int64
dtype: object
```

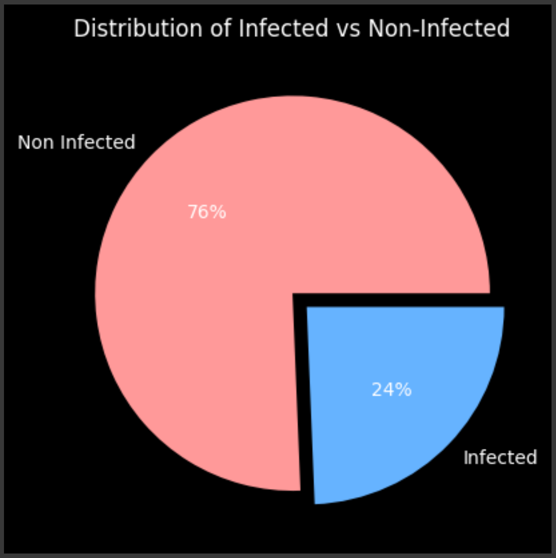
```
def mPlotter(r, c, size, _targets, text):

    bg = '#010108'

    palette = ['#df5337', '#d24644', '#f7d340', '#3339FF', '#440a68', '#84206b', '#f1ef75', '#fbbe23', '#400a67']

    font = 'calibri'

    fig = plt.figure(figsize=size)

    fig.patch.set_facecolor(bg)

    grid = fig.add_gridspec(r, c)

    grid.update(wspace=0.5, hspace=0.25)

    __empty_diff = ((r * c) - 1) - len(_targets)

    axes = []

    for i in range(r):
        for j in range(c):
            axes.append(fig.add_subplot(grid[i, j]))

    for idx, ax in enumerate(axes):
        ax.set_facecolor(bg)

        if idx == 0:
            ax.spines["bottom"].set_visible(False)
            ax.tick_params(left=False, bottom=False)
            ax.set_xticklabels([])
            ax.set_yticklabels([])
            ax.text(0.5, 0.5,
                 f'{text}',
                 horizontalalignment='center',
                 verticalalignment='center',
                 fontsize=18,
                 fontweight='bold',
                 fontfamily=font,
                 color="#fff")
        else:
            if (idx - 1) < len(_targets):
                ax.set_title(_targets[idx - 1].capitalize(), fontsize=14, fontweight='bold', fontfamily=font, color="#fff")
                ax.grid(color='#fff', linestyle=':', axis='y', zorder=0,  dashes=(1,5))
                ax.set_xlabel("")
                ax.set_ylabel("")
            else:
                ax.spines["bottom"].set_visible(False)
                ax.tick_params(left=False, bottom=False)
                ax.set_xticklabels([])
                ax.set_yticklabels([])

        ax.spines["left"].set_visible(False)
        ax.spines["top"].set_visible(False)
        ax.spines["right"].set_visible(False)

    def cb(ax):
        ax.set_xlabel("")
        ax.set_ylabel("")

    if __empty_diff > 0:
        axes = axes[:-1*__empty_diff]

    return axes, palette, cb
```

```
target = 'infected'
cont_cols = ['time', 'age', 'wtkg', 'preanti', 'cd40', 'cd420', 'cd80', 'cd820']
dis_cols = list(set(ds.columns) - set([*cont_cols, target]))

len(cont_cols), len(dis_cols)
```

```
(8, 14)
```

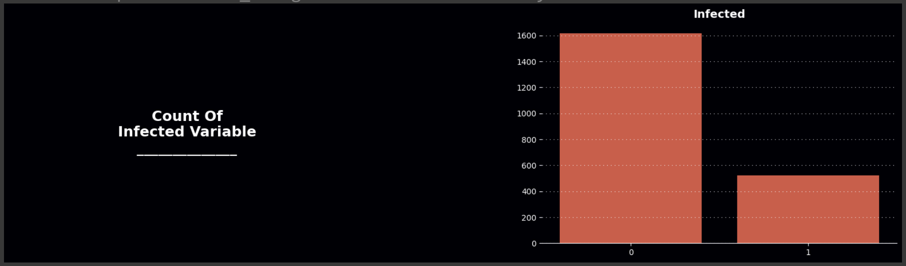
Data Visualization

```
import matplotlib.pyplot as plt

plt.style.use('dark_background')

plt.pie(
    ds['infected'].value_counts(),
    labels=['Non Infected', 'Infected'],
    explode=[0, 0.1],
    autopct='%.0f%%',
    colors=['#ff9999', '#66b3ff']
)

plt.title('Distribution of Infected vs Non-Infected', color='white')

plt.show()
```



```
axes, palette, cb = mPlotter(1, 2, (20, 5), [target], 'Count Of\nInfected Variable\n______________')

sns.countplot(x=ds[target], ax = axes[1], color=palette[0])
cb(axes[1])
```

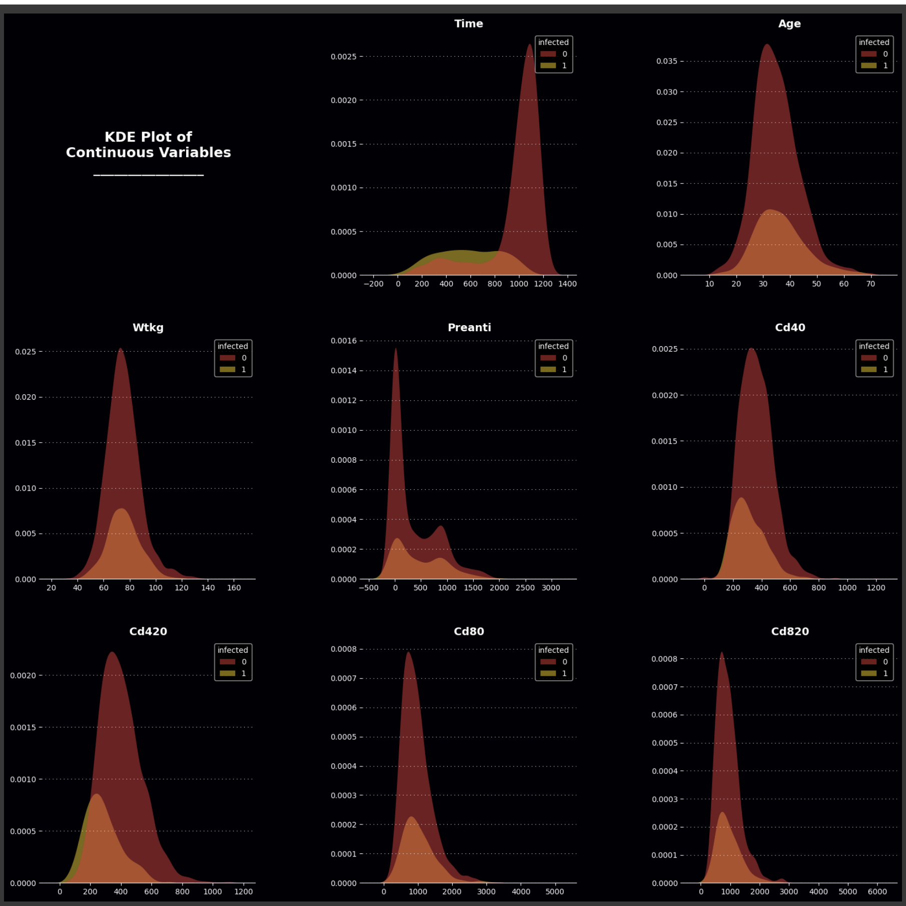
```
WARNING:matplotlib.font_manager:findfont: Font family 'calibri' not found.
WARNING:matplotlib.font_manager:findfont: Font family 'calibri' not found.
WARNING:matplotlib.font_manager:findfont: Font family 'calibri' not found.
WARNING:matplotlib.font_manager:findfont: Font family 'calibri' not found.
WARNING:matplotlib.font_manager:findfont: Font family 'calibri' not found.
WARNING:matplotlib.font_manager:findfont: Font family 'calibri' not found.
WARNING:matplotlib.font_manager:findfont: Font family 'calibri' not found.
WARNING:matplotlib.font_manager:findfont: Font family 'calibri' not found.
WARNING:matplotlib.font_manager:findfont: Font family 'calibri' not found.
WARNING:matplotlib.font_manager:findfont: Font family 'calibri' not found.
WARNING:matplotlib.font_manager:findfont: Font family 'calibri' not found.
WARNING:matplotlib.font_manager:findfont: Font family 'calibri' not found.
WARNING:matplotlib.font_manager:findfont: Font family 'calibri' not found.
WARNING:matplotlib.font_manager:findfont: Font family 'calibri' not found.
WARNING:matplotlib.font_manager:findfont: Font family 'calibri' not found.
WARNING:matplotlib.font_manager:findfont: Font family 'calibri' not found.
WARNING:matplotlib.font_manager:findfont: Font family 'calibri' not found.
WARNING:matplotlib.font_manager:findfont: Font family 'calibri' not found.
WARNING:matplotlib.font_manager:findfont: Font family 'calibri' not found.
WARNING:matplotlib.font_manager:findfont: Font family 'calibri' not found.
```



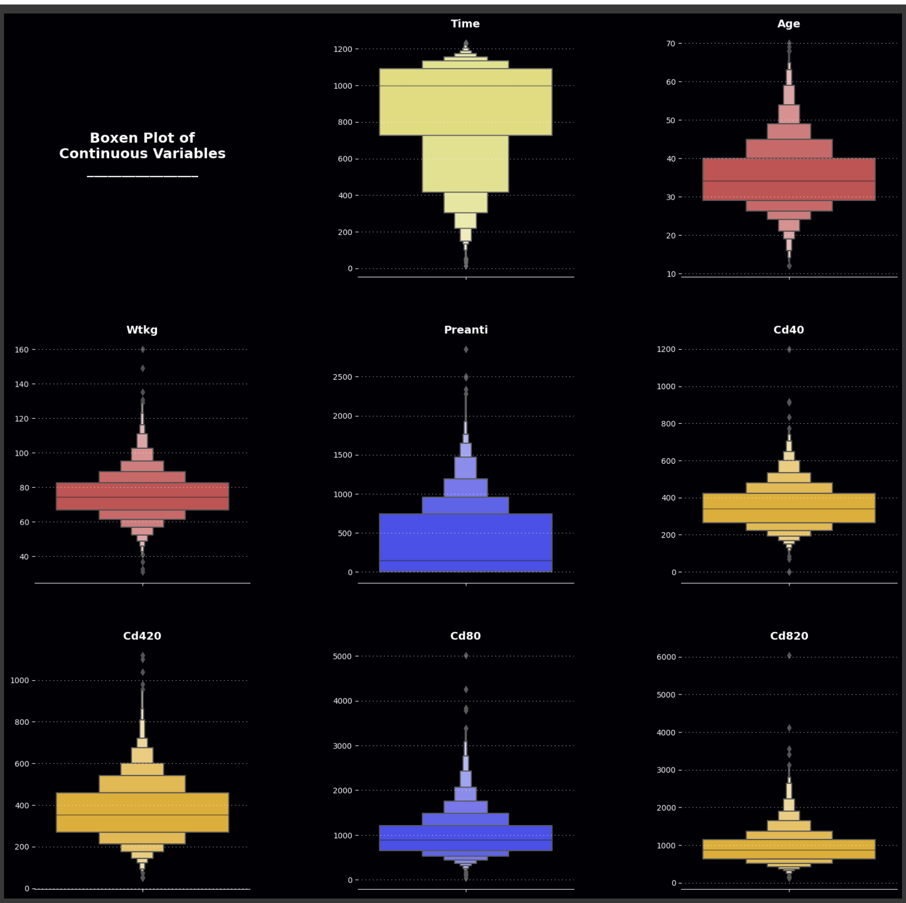
```python
infected_count = ds[target].value_counts()

print(infected_count)
```

```
infected
0    1618
1     521
Name: count, dtype: int64
```

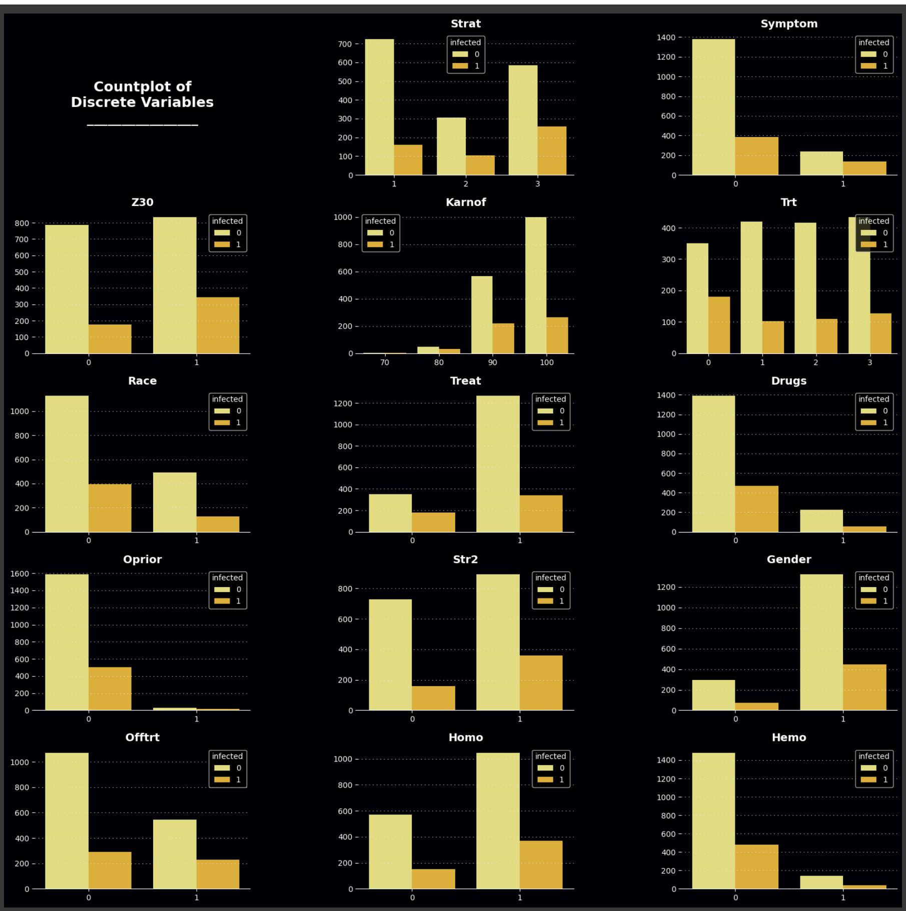
```python
axes, palette, cb = mPlotter(3, 3, (20, 20), cont_cols, 'KDE Plot of\nContinuous Variables\n________________')

for col, ax in zip(cont_cols, axes[1:]):
    sns.kdeplot(data=ds, x=col, ax=ax, hue=target, palette=palette[1:3], alpha=.5, linewidth=0, fill=True)
    cb(ax)
```


KDE Plot of
Continuous Variables
Time
Age
Wtkg
Preanti
Cd40
Cd420
Cd80
Cd820
infected
0
1

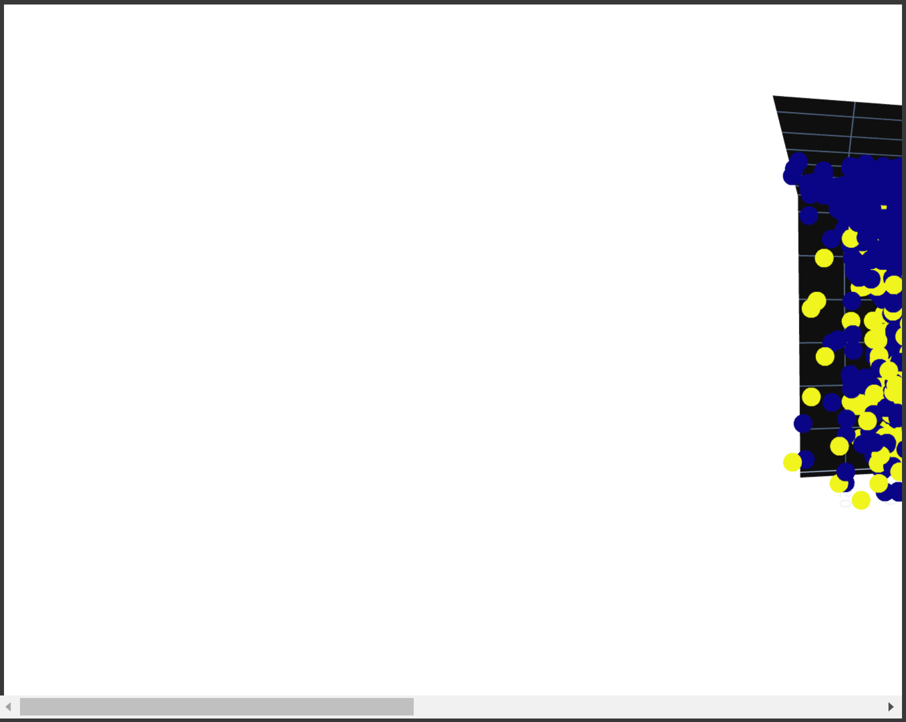
```
axes, palette, cb = mPlotter(3, 3, (20, 20), cont_cols, 'Boxen Plot of\nContinuous Variables\n________________')

for col, ax in zip(cont_cols, axes[1:]):
    sns.boxenplot(data=ds, y=col, ax=ax, palette=[palette[random.randint(0, len(palette)-1)]])
    cb(ax)
```


Boxen Plot of
Continuous Variables
Time
Age
Wtkg
Preanti
Cd40
Cd420
Cd80
Cd820

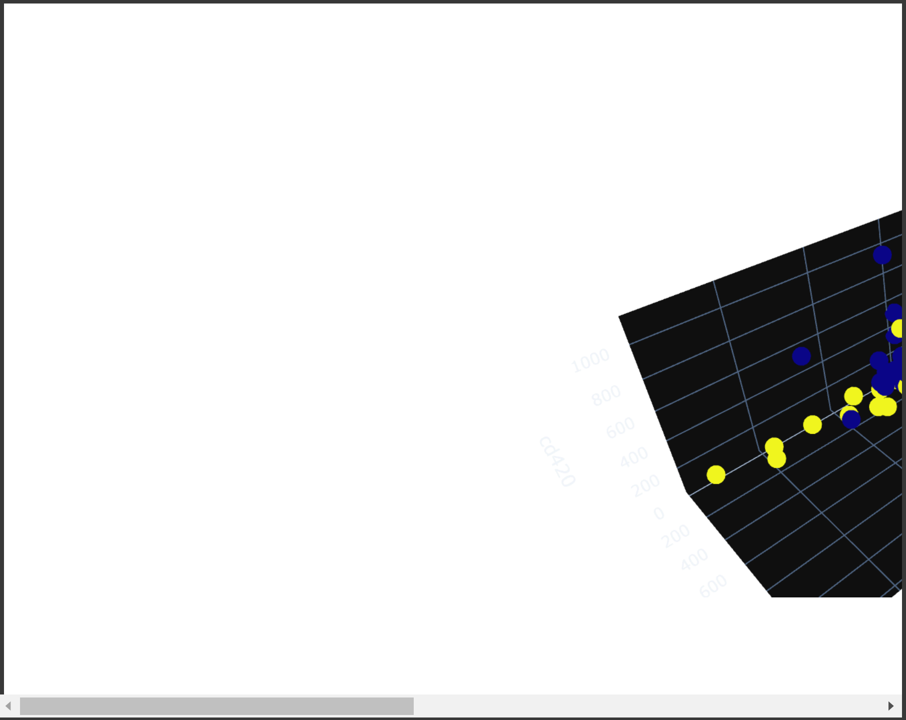
```
axes, palette, cb = mPlotter(5, 3, (20, 20), dis_cols, 'Countplot of\nDiscrete Variables\n________________')

for col, ax in zip(dis_cols, axes[1:]):
    sns.countplot(x=ds[col], ax = ax, hue=ds[target], palette=palette[6:8])
    cb(ax)
```


Countplot of
Discrete Variables
Strat
Symptom
Z30
Karnof
Trt
Race
Treat
Drugs
Oprior
Str2
Gender
Offtrt
Homo
Hemo
infected
0
1

```
ax = px.scatter_3d(ds, x="age", y="wtkg", z="time", template= "plotly_dark", color="infected")

ax.show()
```

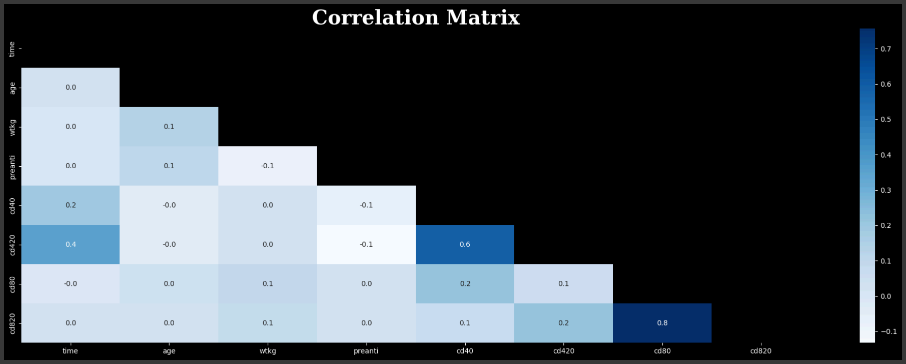


```
ax = px.scatter_3d(ds, x="preanti", y="cd40", z="cd420", template= "plotly_dark", color="infected")

ax.show()
```



```
ax = px.scatter_3d(ds, x="preanti", y="cd80", z="cd820", template= "plotly_dark", color="infected")

ax.show()
```

```
fig = plt.figure(figsize=(25, 8))

gs = fig.add_gridspec(1, 1)

gs.update(wspace=0.3, hspace=0.15)

ax = fig.add_subplot(gs[0, 0])

ax.set_title("Correlation Matrix", fontsize=28, fontweight='bold', fontfamily='serif', color="#fff")

sns.heatmap(ds[cont_cols].corr().transpose(), mask=np.triu(np.ones_like(ds[cont_cols].corr().transpose())), fmt=".1f", annot=True, cmap='Blu

plt.show()
```



```
x_train, x_test, y_train, y_test = train_test_split(ds.iloc[:,:-1], ds.iloc[:, -1], random_state=3, train_size=.7)

x_train.shape, y_train.shape, x_test.shape, y_test.shape
```

```
((1497, 22), (1497,), (642, 22), (642,))
```

Balance the data

```
smote = SMOTE(random_state = 14)

x_train, y_train = smote.fit_resample(x_train, y_train)

x_train.shape, y_train.shape, x_test.shape, y_test.shape
```

```
((2246, 22), (2246,), (642, 22), (642,))
```

Scale the data to ground

```
x_train = MinMaxScaler().fit_transform(x_train)
x_test = MinMaxScaler().fit_transform(x_test)
```

Find best hyperparameter for catboost!

```
def objective(trial):
    params = {
        'iterations': trial.suggest_int('iterations', 100, 1000),
        'learning_rate': trial.suggest_loguniform('learning_rate', 0.01, 0.5),
        'depth': trial.suggest_int('depth', 1, 12),
        'l2_leaf_reg': trial.suggest_loguniform('l2_leaf_reg', 1e-3, 10.0),
        'border_count': trial.suggest_int('border_count', 1, 255),
        'thread_count': -1,
        'loss_function': 'MultiClass',
        'eval_metric': 'Accuracy',
        'verbose': False
    }

    model = CatBoostClassifier(**params)

    model.fit(x_train, y_train, eval_set=(x_test, y_test), verbose=False, early_stopping_rounds=20)

    y_pred = model.predict(x_test)

    accuracy = accuracy_score(y_test, y_pred)

    return accuracy

study = optuna.create_study(direction='maximize')

study.optimize(objective, n_trials=50, show_progress_bar=True)
```

```
[I 2024-04-27 11:30:39,199] A new study created in memory with name: no-name-af863032-91
  0%|          | 0/50 [00:00<?, ?it/s]
[I 2024-04-27 11:30:39,449] Trial 0 finished with value: 0.8255451713395638 and paramete
[I 2024-04-27 11:30:39,622] Trial 1 finished with value: 0.897196261682243 and parameter
[I 2024-04-27 11:30:39,714] Trial 2 finished with value: 0.897196261682243 and parameter
[I 2024-04-27 11:30:44,736] Trial 3 finished with value: 0.8987538940809969 and paramete
[I 2024-04-27 11:30:44,870] Trial 4 finished with value: 0.883177570093458 and parameter
[I 2024-04-27 11:30:45,796] Trial 5 finished with value: 0.8956386292834891 and paramete
[I 2024-04-27 11:30:45,980] Trial 6 finished with value: 0.8878504672897196 and paramete
[I 2024-04-27 11:30:46,473] Trial 7 finished with value: 0.8925233644859814 and paramete
[I 2024-04-27 11:30:46,583] Trial 8 finished with value: 0.9018691588785047 and paramete
[I 2024-04-27 11:30:47,362] Trial 9 finished with value: 0.9065420560747663 and paramete
[I 2024-04-27 11:30:48,653] Trial 10 finished with value: 0.8925233644859814 and paramet
[I 2024-04-27 11:30:48,897] Trial 11 finished with value: 0.9049844236760125 and paramet
[I 2024-04-27 11:30:49,134] Trial 12 finished with value: 0.9080996884735203 and paramet
[I 2024-04-27 11:30:49,334] Trial 13 finished with value: 0.8753894080996885 and paramet
[I 2024-04-27 11:30:50,353] Trial 14 finished with value: 0.897196261682243 and paramete
[I 2024-04-27 11:30:50,677] Trial 15 finished with value: 0.9018691588785047 and paramet
[I 2024-04-27 11:30:50,946] Trial 16 finished with value: 0.9034267912772586 and paramet
[I 2024-04-27 11:30:52,907] Trial 17 finished with value: 0.8894080996884736 and paramet
[I 2024-04-27 11:30:53,070] Trial 18 finished with value: 0.8878504672897196 and paramet
[I 2024-04-27 11:30:53,677] Trial 19 finished with value: 0.9034267912772586 and paramet
[I 2024-04-27 11:30:54,563] Trial 20 finished with value: 0.8940809968847352 and paramet
[I 2024-04-27 11:30:54,710] Trial 21 finished with value: 0.881619937694704 and paramete
[I 2024-04-27 11:30:54,918] Trial 22 finished with value: 0.897196261682243 and paramete
[I 2024-04-27 11:30:55,035] Trial 23 finished with value: 0.8302180685358256 and paramet
[I 2024-04-27 11:30:55,340] Trial 24 finished with value: 0.9049844236760125 and paramet
[I 2024-04-27 11:30:55,492] Trial 25 finished with value: 0.8909657320872274 and paramet
[I 2024-04-27 11:30:56,113] Trial 26 finished with value: 0.9003115264797508 and paramet
[I 2024-04-27 11:30:56,284] Trial 27 finished with value: 0.8878504672897196 and paramet
[I 2024-04-27 11:30:56,630] Trial 28 finished with value: 0.9034267912772586 and paramet
[I 2024-04-27 11:30:56,749] Trial 29 finished with value: 0.8193146417445483 and paramet
[I 2024-04-27 11:30:56,973] Trial 30 finished with value: 0.8847352024922118 and paramet
[I 2024-04-27 11:30:57,257] Trial 31 finished with value: 0.9065420560747663 and paramet
[I 2024-04-27 11:30:57,496] Trial 32 finished with value: 0.9065420560747663 and paramet
[I 2024-04-27 11:30:57,731] Trial 33 finished with value: 0.9065420560747663 and paramet
[I 2024-04-27 11:30:57,956] Trial 34 finished with value: 0.8894080996884736 and paramet
[I 2024-04-27 11:30:58,190] Trial 35 finished with value: 0.9049844236760125 and paramet
[I 2024-04-27 11:30:58,607] Trial 36 finished with value: 0.8878504672897196 and paramet
[I 2024-04-27 11:30:58,772] Trial 37 finished with value: 0.9018691588785047 and paramet
[I 2024-04-27 11:30:58,990] Trial 38 finished with value: 0.8987538940809969 and paramet
[I 2024-04-27 11:30:59,145] Trial 39 finished with value: 0.9034267912772586 and paramet
[I 2024-04-27 11:30:59,362] Trial 40 finished with value: 0.8987538940809969 and paramet
[I 2024-04-27 11:30:59,581] Trial 41 finished with value: 0.9049844236760125 and paramet
[I 2024-04-27 11:30:59,737] Trial 42 finished with value: 0.8956386292834891 and paramet
[I 2024-04-27 11:31:00,071] Trial 43 finished with value: 0.9003115264797508 and paramet
[I 2024-04-27 11:31:00,286] Trial 44 finished with value: 0.9080996884735203 and paramet
[I 2024-04-27 11:31:00,560] Trial 45 finished with value: 0.9018691588785047 and paramet
[I 2024-04-27 11:31:01,943] Trial 46 finished with value: 0.8987538940809969 and paramet
[I 2024-04-27 11:31:02,288] Trial 47 finished with value: 0.9003115264797508 and paramet
[I 2024-04-27 11:31:02,581] Trial 48 finished with value: 0.9034267912772586 and paramet
[I 2024-04-27 11:31:02,742] Trial 49 finished with value: 0.8925233644859814 and paramet
```

```
model = CatBoostClassifier(
    verbose=0,
    random_state=3,
    **study.best_params
)

model.fit(x_train, y_train)

y_pred = model.predict(x_test)
```

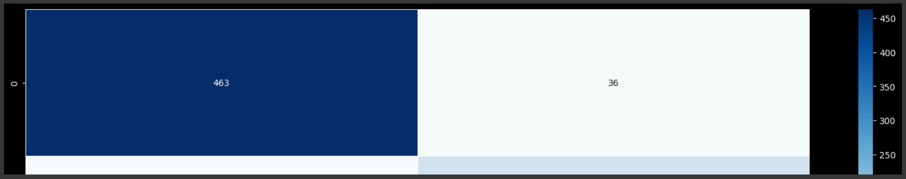
Oversee the model performance

```
print (classification_report(y_pred, y_test))
```

```
              precision    recall  f1-score   support

           0       0.94      0.93      0.93       499
           1       0.76      0.78      0.77       143

    accuracy                           0.89       642
   macro avg       0.85      0.85      0.85       642
weighted avg       0.90      0.89      0.89       642
```

```
plt.subplots(figsize=(20, 6))

sns.heatmap(confusion_matrix(y_pred, y_test), annot = True, fmt="d", cmap="Blues", linewidths=.5)
```

```
<Axes: >
```


0
463
36
450
400
350
300
250