1. Install Required Libraries

```
###!pip install -q comet_ml gradio
```

```
from comet_ml import Experiment
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.feature_extraction.text import TfidfTransformer
from sklearn.pipeline import Pipeline
from sklearn.datasets import fetch_20newsgroups
from sklearn.linear_model import SGDClassifier
from sklearn.metrics import accuracy_score
from sklearn.model_selection import KFold
```

```
import comet_ml
```

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
```

```
###!mkdir ~/.kaggle
```

```
###! pip install kaggle
```

```
###!cp /kaggle.json ~/.kaggle/
```

```
###!chmod 600 ~/.kaggle/kaggle.json
```

```
###!pip install keras-tuner
```

```
###!kaggle datasets download -d muhammadehsan000/diabetes-healthcare-dataset
```

```
###! unzip /content/diabetes-healthcare-dataset.zip
```

```
Archive:  /content/diabetes-healthcare-dataset.zip
  inflating: Diabetes-Data.csv
```

```
experiment = Experiment(api_key="u4v1dA5tEc5tOx0euTnHNMnDs",
                        project_name="ml-test",
                        workspace="debmalyaray9989",
                       )
```

```
COMET WARNING: As you are running in a Jupyter environment, you will need to call `experiment.end()` when finished to
COMET INFO: Experiment is live on comet.com https://www.comet.com/debmalyaray9989/ml-test/d0d4ee16d00842ad981bda90152d

COMET INFO: Couldn't find a Git repository in '/content' nor in any parent directory. Set `COMET_GIT_DIRECTORY` if you
```

```
diabetes_data = pd.read_csv('/content/Diabetes-Data.csv')
diabetes_data.head(5)
```

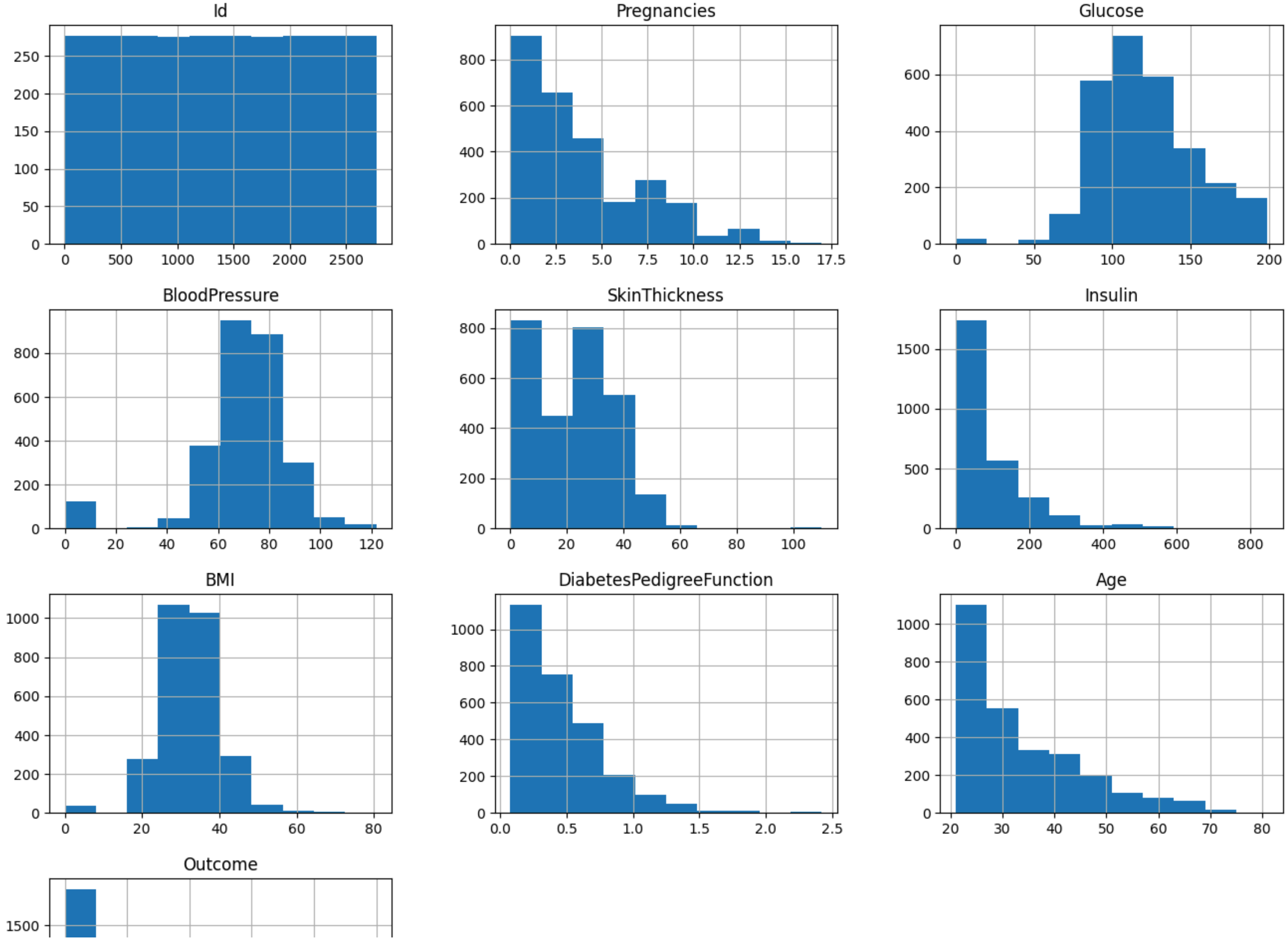
	Id	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	1	6	148	72	35	0	33.6	0.627	50	1
1	2	1	85	66	29	0	26.6	0.351	31	0
2	3	8	183	64	0	0	23.3	0.672	32	1
3	4	1	89	66	23	94	28.1	0.167	21	0
4	5	0	137	40	35	168	43.1	2.288	33	1

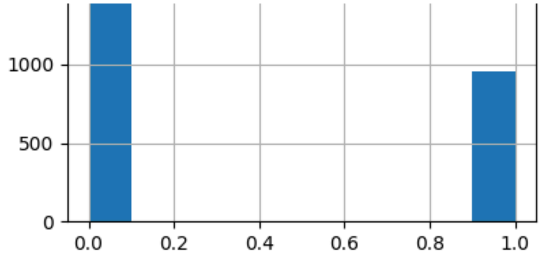
```
diabetes_data.columns
```

```
Index(['Id', 'Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness',
       'Insulin', 'BMI', 'DiabetesPedigreeFunction', 'Age', 'Outcome'],
      dtype='object')
```

```
diabetes_data.hist(figsize=(16,14));
experiment.log_figure(figure=plt)
```

```
{'web': 'https://www.comet.com/api/image/download?imageId=82db7cde4f5941a5addeeadae252d7e1&experimentKey=d0d4ee16d00842ad981bda90152d8864',
 'api': 'https://www.comet.com/api/rest/v1/image/get-image?imageId=82db7cde4f5941a5addeeadae252d7e1&experimentKey=d0d4ee16d00842ad981bda90152d8864',
 'imageId': '82db7cde4f5941a5addeeadae252d7e1'}
```

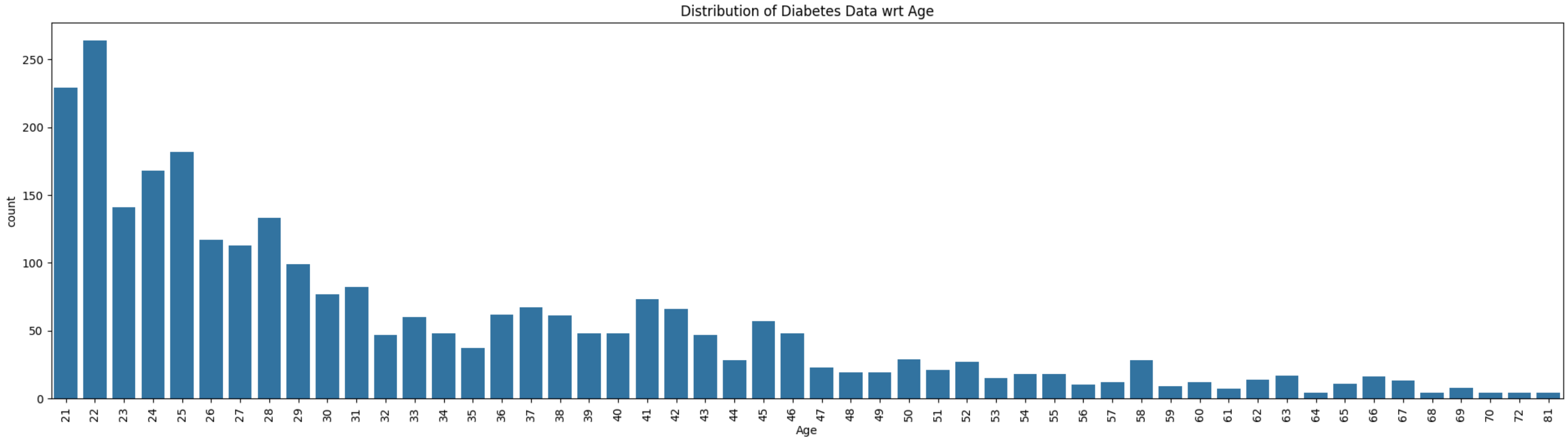




```
diabetes_data.Age.unique()
```

```
array([50, 31, 32, 21, 33, 30, 26, 29, 53, 54, 34, 57, 59, 51, 27, 41, 43,
       22, 38, 60, 28, 45, 35, 46, 56, 37, 48, 40, 25, 24, 58, 42, 44, 39,
       36, 23, 61, 69, 62, 55, 65, 47, 52, 66, 49, 63, 67, 72, 81, 64, 70,
       68])
```

```
plt.figure(figsize=(24,6))
plt.xticks(rotation=90)
ax = sns.countplot(x=diabetes_data.Age)
ax.set_title("Distribution of Diabetes Data wrt Age")
experiment.log_figure(figure=plt)
plt.show()
```



```
###! pip install klib
```

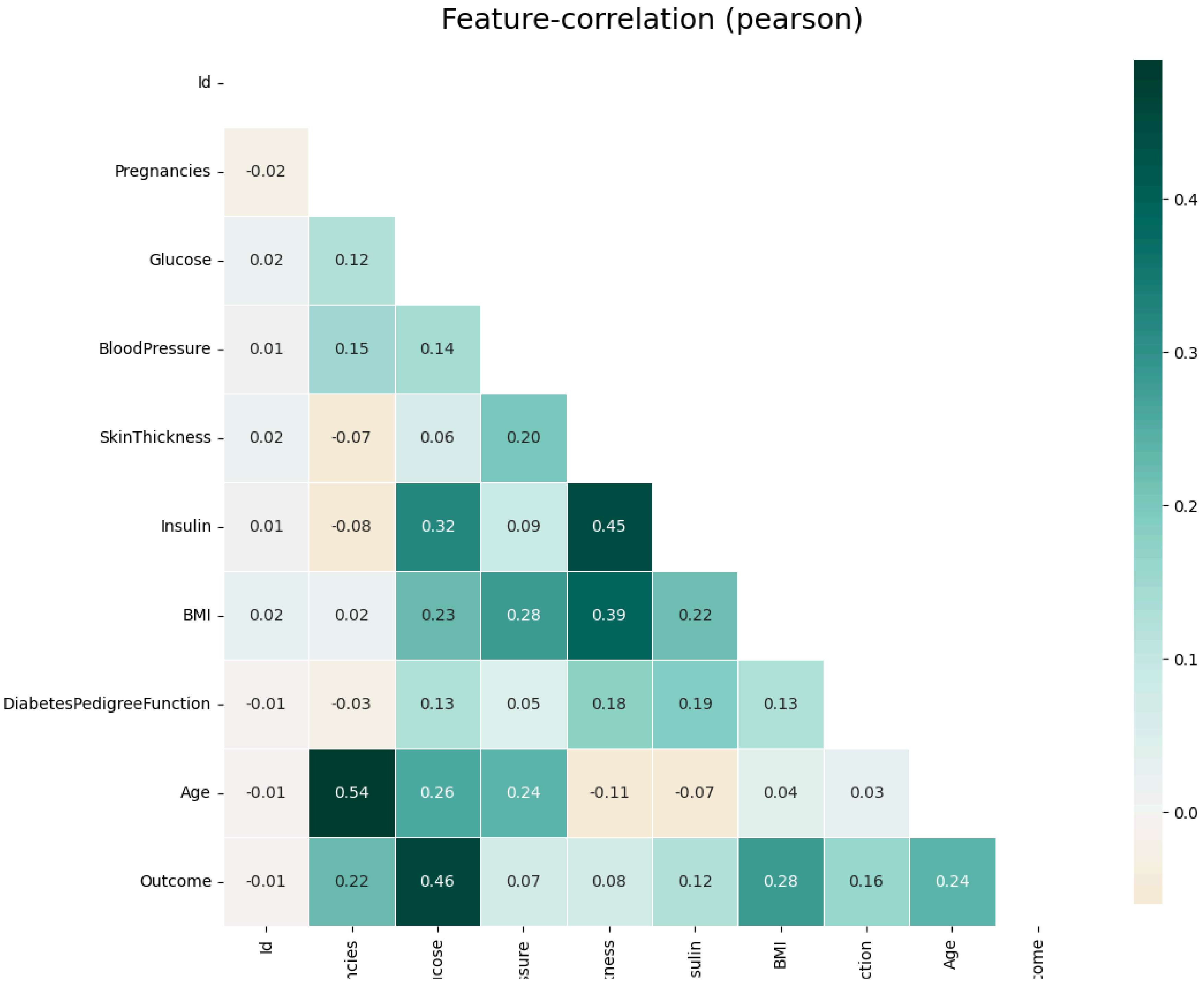
```
import klib
```

```
klib.cat_plot(diabetes_data)
```

```
No columns with categorical data were detected.
```

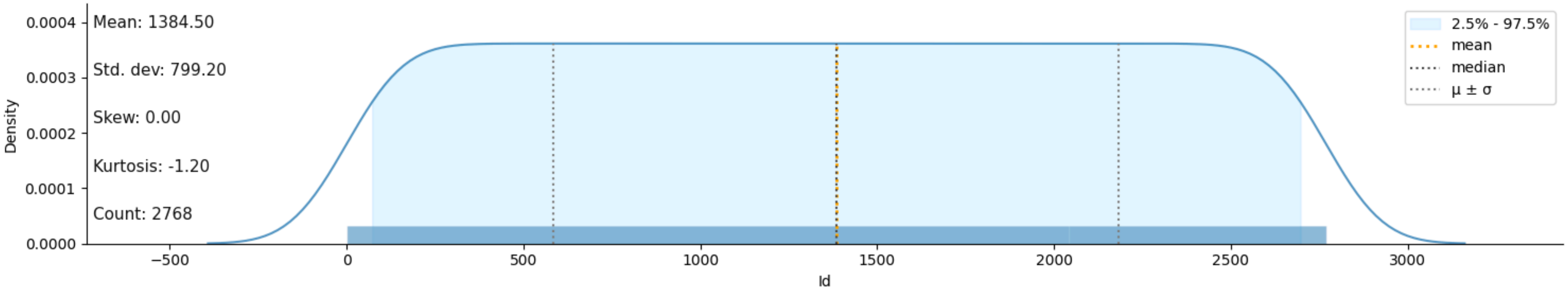
```
plt.figure(figsize=(24,6))
klib.corr_plot(diabetes_data)
experiment.log_figure(figure=plt)
plt.show()
```

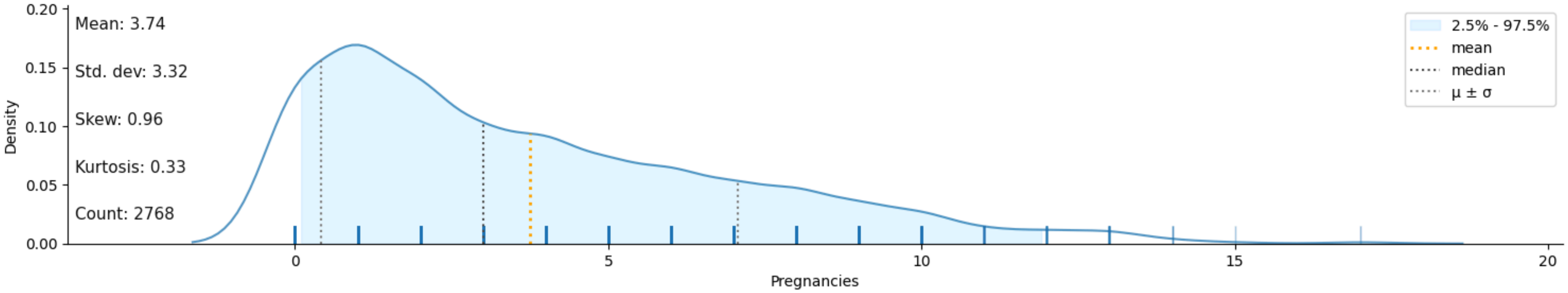
```
<Figure size 2400x600 with 0 Axes>
```

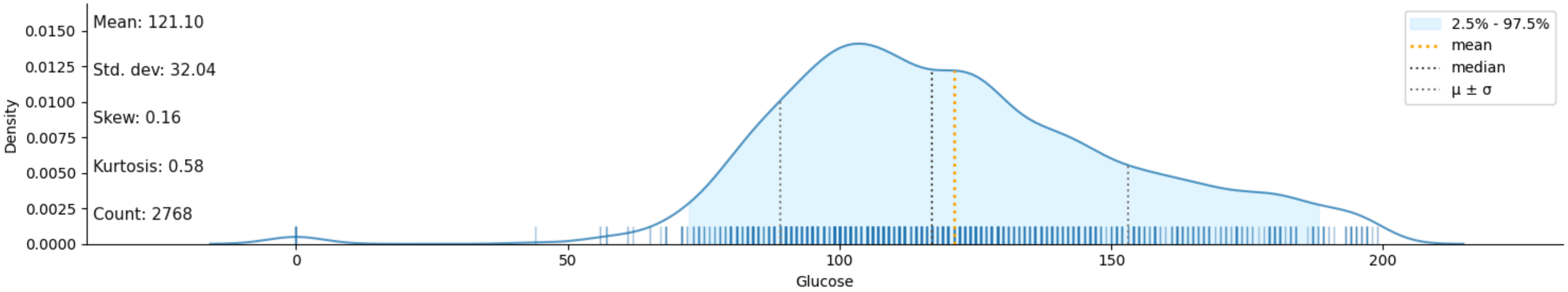

Feature-correlation (pearson)
Id
Pregnancies -0.02
Glucose 0.02 0.12
BloodPressure 0.01 0.15 0.14
SkinThickness 0.02 -0.07 0.06 0.20
Insulin 0.01 -0.08 0.32 0.09 0.45
BMI 0.02 0.02 0.23 0.28 0.39 0.22
DiabetesPedigreeFunction -0.01 -0.03 0.13 0.05 0.18 0.19 0.13
Age -0.01 0.54 0.26 0.24 -0.11 -0.07 0.04 0.03
Outcome -0.01 0.22 0.46 0.07 0.08 0.12 0.28 0.16 0.24
Id
ncies
icose
ssure
ness
sulin
BMI
ction
Age
come
0.4
0.3
0.2
0.1
0.0

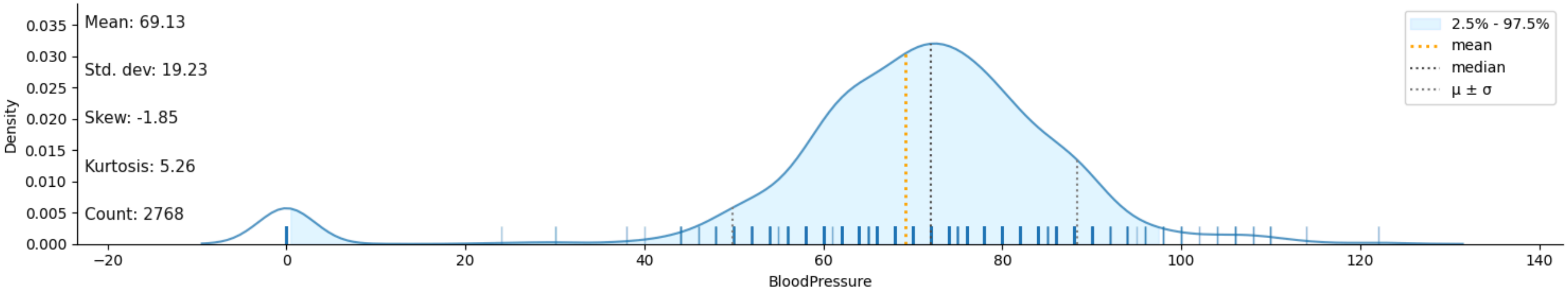
```
plt.figure(figsize=(24,6))
klib.dist_plot(diabetes_data)
experiment.log_figure(figure=plt)
plt.show()
```

```
<Figure size 2400x600 with 0 Axes>
```

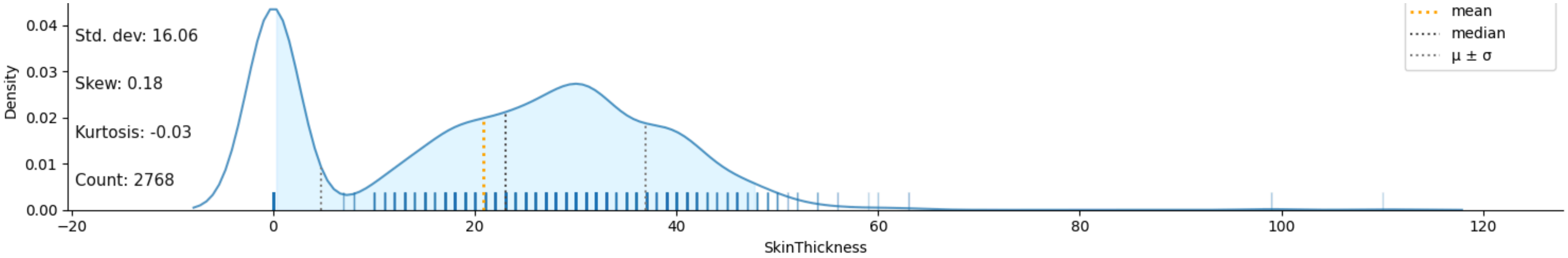

Mean: 1384.50
Std. dev: 799.20
Skew: 0.00
Kurtosis: -1.20
Count: 2768
Density
0.0004
0.0003
0.0002
0.0001
0.0000
−500
0
500
1000
1500
2000
2500
3000
Id
2.5% - 97.5%
mean
median
μ ± σ

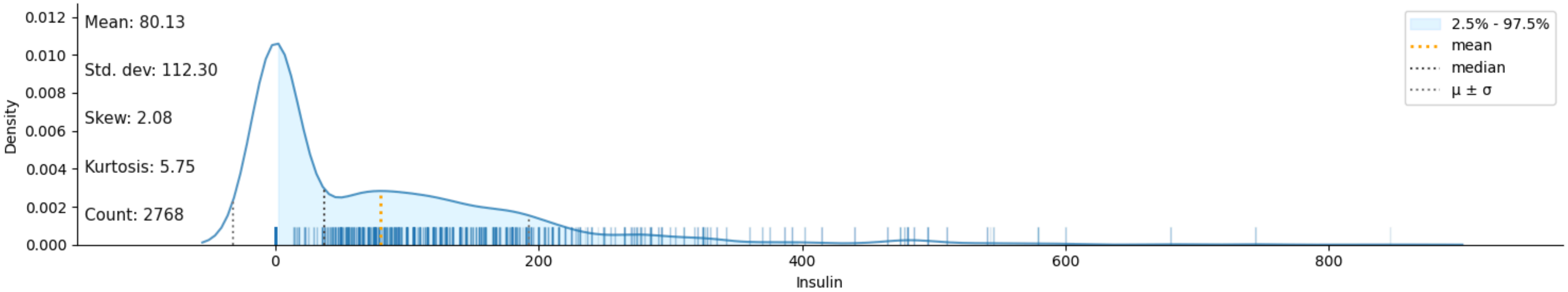

Mean: 3.74
Std. dev: 3.32
Skew: 0.96
Kurtosis: 0.33
Count: 2768
Density
0.20
0.15
0.10
0.05
0.00
0
5
10
15
20
Pregnancies
2.5% - 97.5%
mean
median
μ ± σ

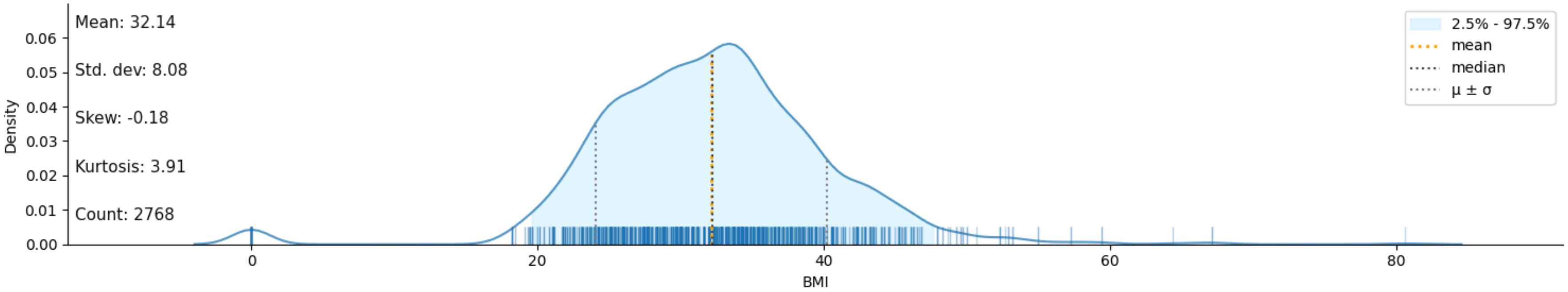

Mean: 121.10
Std. dev: 32.04
Skew: 0.16
Kurtosis: 0.58
Count: 2768
Density
0.0150
0.0125
0.0100
0.0075
0.0050
0.0025
0.0000
0
50
100
150
200
Glucose
2.5% - 97.5%
mean
median
μ ± σ

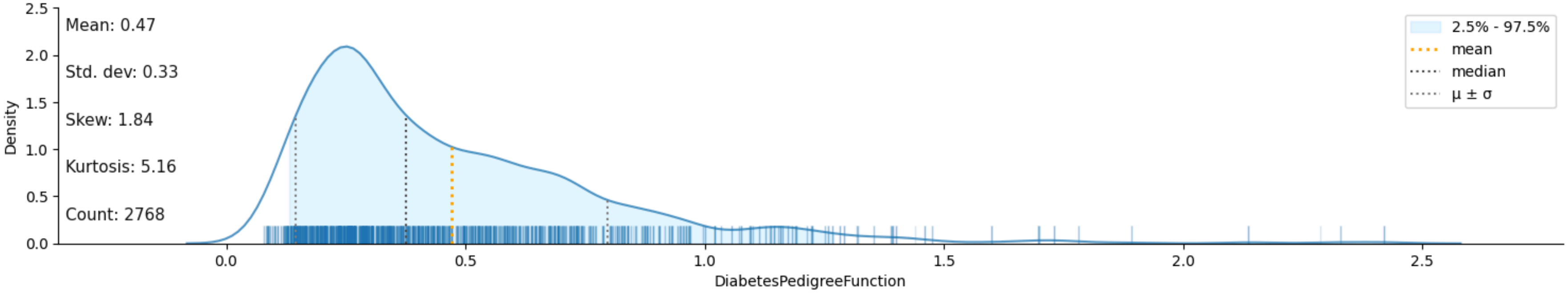

Mean: 69.13
Std. dev: 19.23
Skew: -1.85
Kurtosis: 5.26
Count: 2768
Density
0.035
0.030
0.025
0.020
0.015
0.010
0.005
0.000
−20
0
20
40
60
80
100
120
140
BloodPressure
2.5% - 97.5%
mean
median
μ ± σ

0.05
Mean: 20.82
2.5% - 97.5%

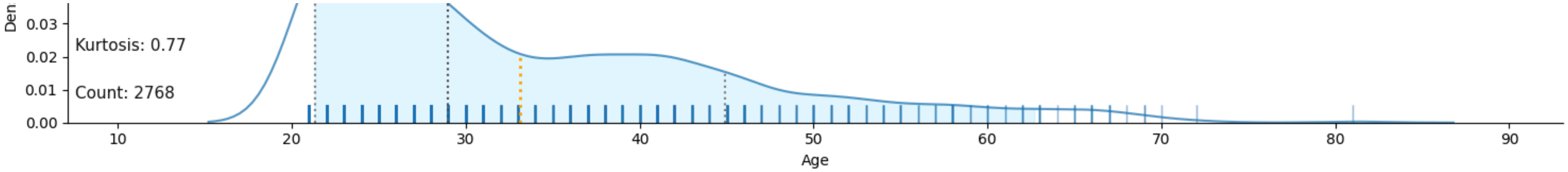

Std. dev: 16.06
Skew: 0.18
Kurtosis: -0.03
Count: 2768
Density
0.04
0.03
0.02
0.01
0.00
−20
0
20
40
60
80
100
120
SkinThickness
mean
median
μ ± σ


Mean: 80.13
Std. dev: 112.30
Skew: 2.08
Kurtosis: 5.75
Count: 2768
Density
0.012
0.010
0.008
0.006
0.004
0.002
0.000
0
200
400
600
800
Insulin
2.5% - 97.5%
mean
median
μ ± σ


Mean: 32.14
Std. dev: 8.08
Skew: -0.18
Kurtosis: 3.91
Count: 2768
Density
0.06
0.05
0.04
0.03
0.02
0.01
0.00
0
20
40
60
80
BMI
2.5% - 97.5%
mean
median
μ ± σ


Mean: 0.47
Std. dev: 0.33
Skew: 1.84
Kurtosis: 5.16
Count: 2768
Density
2.5
2.0
1.5
1.0
0.5
0.0
0.0
0.5
1.0
1.5
2.0
2.5
DiabetesPedigreeFunction
2.5% - 97.5%
mean
median
μ ± σ

Mean: 33.13
Std. dev: 11.78
Skew: 1.17
sity
0.07
0.06
0.05
0.04
2.5% - 97.5%
mean
median
μ ± σ


Den
0.03
0.02
0.01
0.00
Kurtosis: 0.77
Count: 2768
10
20
30
40
50
60
70
80
90
Age

```
plt.figure(figsize=(24,6))
klib.corr_mat(diabetes_data)
experiment.log_figure(figure=plt)
plt.show()
```

```
COMET WARNING: Refuse to upload empty figure, please call log_figure before calling show
COMET WARNING: Failing to save the matplotlib figure
<Figure size 2400x600 with 0 Axes>
```

```
klib.missingval_plot(diabetes_data)
```

```
No missing values found in the dataset.
```

```
diabetes_data.dtypes
```

	0
Id	int64
Pregnancies	int64
Glucose	int64
BloodPressure	int64
SkinThickness	int64
Insulin	int64
BMI	float64
DiabetesPedigreeFunction	float64
Age	int64
Outcome	int64

dtype: object

```
from sklearn.model_selection import train_test_split
```

```
X = diabetes_data.drop(columns="DiabetesPedigreeFunction")
y = diabetes_data['DiabetesPedigreeFunction']
```

```
y = pd.DataFrame(y)
```

```
print(X.columns)
print(y.columns)
```

```
Index(['Id', 'Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness',
       'Insulin', 'BMI', 'Age', 'Outcome'],
      dtype='object')
Index(['DiabetesPedigreeFunction'], dtype='object')
```

```
X_train, X_test, y_train, y_test = train_test_split(
  X, y , random_state=104,test_size=0.25, shuffle=True)
```

```
print(X_train.shape)
print(X_test.shape)
print(y_train.shape)
print(y_test.shape)
```

```
(2076, 9)
(692, 9)
(2076, 1)
(692, 1)
```

```
#Specifing the hyperparameters we want to tune in our algorithm
model_params ={
    "n_estimators": {
            "type": "discrete",
            "values": [11, 12, 13]
    },
    "max_depth": {
            "type": "discrete",
            "values": [3, 4, 5]
    },
    "learning_rate": {
            "type": "discrete",
            "values": [0.05, 0.1, 0.2]
    },
    "min_child_weight": {
            "type": "discrete",
            "values": [1, 2, 3]
    },
    "subsample": {
            "type": "discrete",
            "values": [0.8, 0.9, 1]
    }
}
```

```
# Specifing the parameters with want to supply to the optimizer config
optimizer_dict= {
    "algorithm": "random",
    "trials": 1,
    "parameters": model_params,
    "name": "My Random Search",
}


# Initializing our optimizer
opt = comet_ml.Optimizer(api_key="u4v1dA5tEc5tOx0euTnHNMnDs",  config=optimizer_dict)
```

```
COMET INFO: 562b52b34b284c998fef05ce2346fc3b
COMET INFO: Using optimizer config: {'algorithm': 'random', 'configSpaceSize': 243, 'endTime': None, 'id': '562b52b34b
```

```
###! pip install xgboost
```

```
from xgboost import XGBRegressor
```

```
for experiment in opt.get_experiments(project_name="Tree-based ML-Optimize"):
    # Initializing XGBoost
    # Passing the each paramter to our model by using the get_parameter method from experiment
    model = XGBRegressor(
        n_estimators=experiment.get_parameter("n_estimators"),
        max_depth=experiment.get_parameter("max_depth"),
        learning_rate=experiment.get_parameter("learning_rate"),
        min_child_weight=experiment.get_parameter("min_child_weight"),
        subsample=experiment.get_parameter("subsample"),
        random_state=42)
```

```
COMET INFO:     installed packages  : 1
COMET INFO:     notebook            : 1
COMET INFO:     os packages         : 1
COMET INFO:     source_code         : 1
COMET INFO:
COMET WARNING: As you are running in a Jupyter environment, you will need to call `experiment.end()` when finished t
COMET INFO: Experiment is live on comet.com https://www.comet.com/debmalyaray9989/tree-based-ml-optimize/24d521c7fd5

COMET INFO: Couldn't find a Git repository in '/content' nor in any parent directory. Set `COMET_GIT_DIRECTORY` if y
```

```
#plt.figure(figsize=(24,8))
#plt.bar(range(len(model.feature_importances_)), model.feature_importances_)
#plt.xticks(range(len(model.feature_importances_)), X_train.columns)
#experiment.log_figure(figure=plt)
#plt.show()
```

```
# Training the model with the training set.
my_pipeline = Pipeline(steps=[('model', model)])
my_pipeline.fit(X_train,y_train)
```

```
▸ Pipeline
  ▸ XGBRegressor
```

```
from sklearn.metrics import confusion_matrix, f1_score, precision_score, recall_score, r2_score
```

```
# Calculating the r2 score on the validation data
y_test_pred = my_pipeline.predict(X_test)
r2_val = np.round(r2_score(y_test, y_test_pred),2)
```

```
# Calculating the r2 score on the training data
y_train_pred= my_pipeline.predict(X_train)
r2_train = np.round(r2_score(y_train, y_train_pred),2)
```