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
```

```
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 ensure all metrics and code ar
COMET INFO: Experiment is live on comet.com https://www.comet.com/debmalyaray9989/ml-test/349a3b144a2c4559bd16721d43c87df3
```

```
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

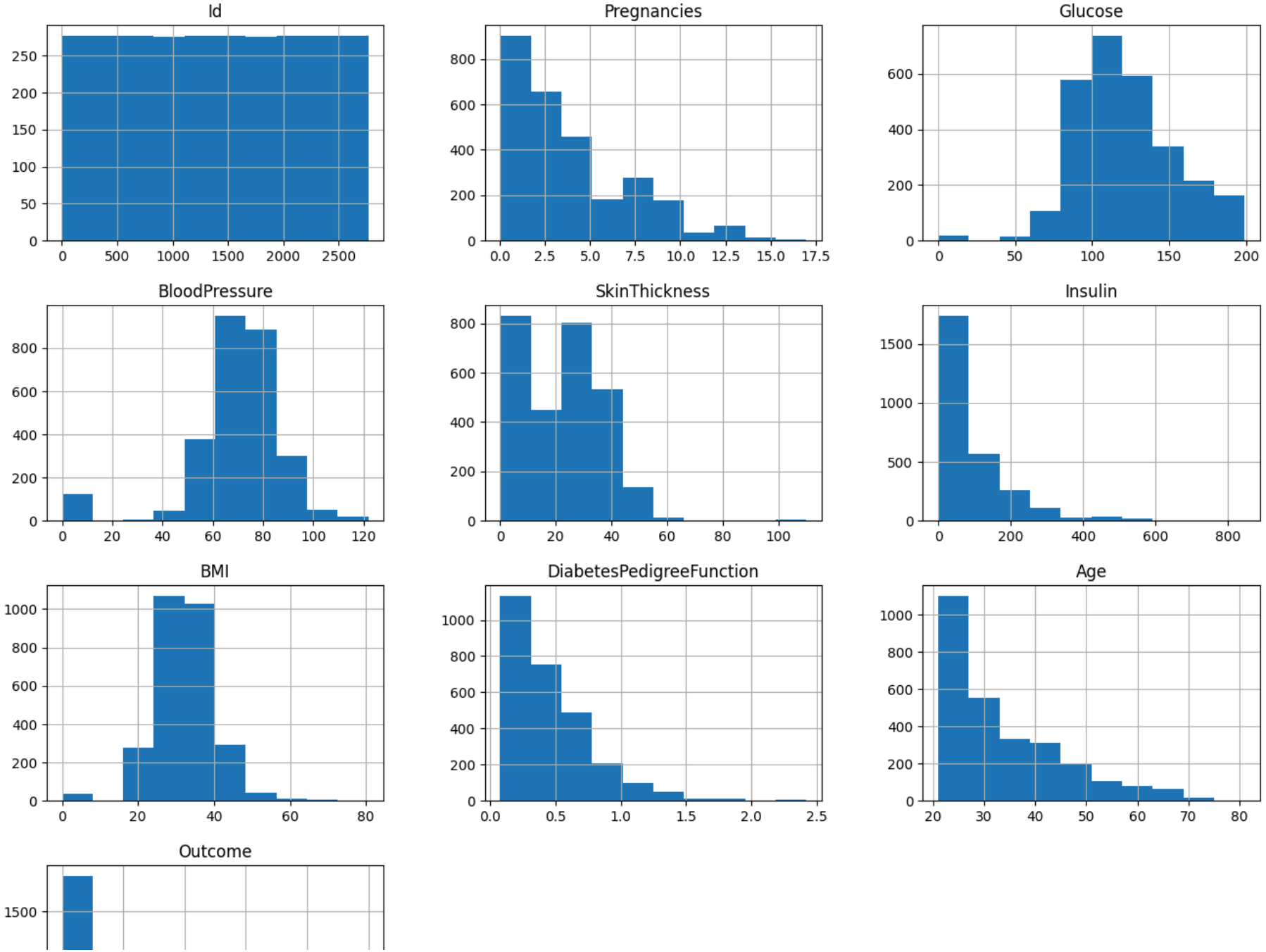
Next steps: Generate code with diabetes_data | View recommended plots | New interactive sheet

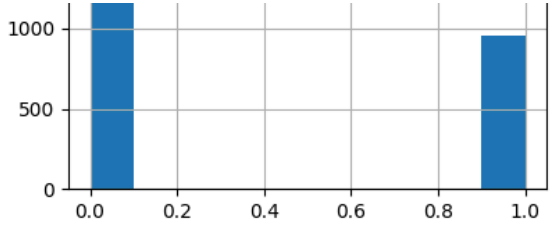
```
diabetes_data.columns
```

```
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=56d5f65a981848e4beba1830d2761ef8&experimentKey=349a3b144a2c4559bd16721d43c87df3',
'api': 'https://www.comet.com/api/rest/v1/image/get-image?imageId=56d5f65a981848e4beba1830d2761ef8&experimentKey=349a3b144a2c4559bd16721d43c87df3',
'imageId': '56d5f65a981848e4beba1830d2761ef8'}

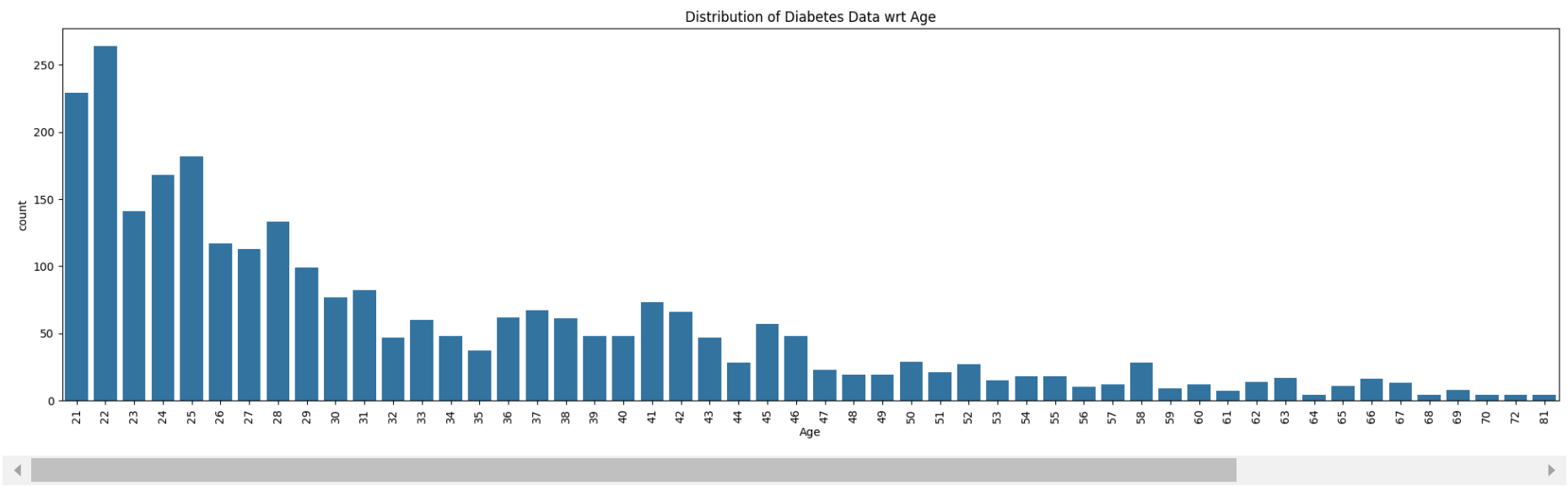





```
diabetes_data.Age.unique()
```

```
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()
```




```
diabetes_data.dtypes
```

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

```
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",
}
```

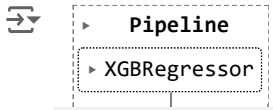
```
# Initializing our optimizer
opt = comet_ml.Optimizer(api_key="u4v1dA5tEc5tOx0euTnHNMnDs",  config=optimizer_dict)
```

```
COMET INFO: 6669769cbe0b43578f4b6d5279f40422
COMET INFO: Using optimizer config: {'algorithm': 'random', 'configSpaceSize': 243, 'endTime': None, 'id': '6669769cbe0b43578f4b6d5279f40422', 'last
```

```
###! 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:     optimizer_process      : 21291
COMET INFO:     optimizer_trial        : 1
COMET INFO:     optimizer_version      : 2.0.26
COMET INFO:   Parameters:
COMET INFO:     learning_rate    : 0.1
COMET INFO:     max_depth        : 4
COMET INFO:     min_child_weight : 2
COMET INFO:     n_estimators     : 11
COMET INFO:     subsample        : 1
COMET INFO:   Uploads:
COMET INFO:     environment details : 1
COMET INFO:     filename            : 1
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 to ensure all metrics and code
COMET INFO: Experiment is live on comet.com https://www.comet.com/debmalyaray9989/tree-based-ml-optimize/87824afbb5c741ebb26cd2e881f90335

COMET INFO: Couldn't find a Git repository in '/content' nor in any parent directory. Set `COMET_GIT_DIRECTORY` if your Git Repository is elsewhe
COMET INFO: Optimizer search 6669769cbe0b43578f4b6d5279f40422 has completed
```

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




```
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)
```

```
# logging the metrics to the comet website
experiment.log_parameter("random_state", 42)
experiment.log_metrics({"r2_validation": np.round(r2_val,2),
                        "r2_train": np.round(r2_train,2)})


from sklearn.metrics import median_absolute_error, mean_squared_error


mae = mean_squared_error(y_test, y_test_pred)


metrics = {"mean_squared_error":mae}


experiment.log_metrics(metrics)


###! pip install joblib


import joblib


joblib.dump(my_pipeline, 'my_pipeline_model.joblib')
```

```
['my_pipeline_model.joblib']
```

```
### u4v1dA5tEc5tOx0euTnHNMnDs


from comet_ml import Experiment
exp = Experiment(api_key="u4v1dA5tEc5tOx0euTnHNMnDs")
exp.log_model("my_pipeline_model", "/content/my_pipeline_model.joblib")
```

```
COMET INFO:     model_missing                  : nan
COMET INFO:     model_monotone_constraints     : None
COMET INFO:     model_multi_strategy           : None
COMET INFO:     model_n_estimators             : 11
COMET INFO:     model_n_jobs                   : None
COMET INFO:     model_num_parallel_tree        : None
COMET INFO:     model_objective                : reg:squarederror
COMET INFO:     model_random_state             : 42
COMET INFO:     model_reg_alpha                : None
COMET INFO:     model_reg_lambda               : None
COMET INFO:     model_sampling_method          : None
COMET INFO:     model_scale_pos_weight         : None
COMET INFO:     model_subsample                : 0.8
COMET INFO:     model_tree_method              : None
COMET INFO:     model_validate_parameters      : None
COMET INFO:     model_verbosity                : None
COMET INFO:     n_estimators                   : 11
COMET INFO:     objective                      : reg:squarederror
COMET INFO:     random_state                   : 42
COMET INFO:     rank                           : 0
COMET INFO:     subsample                      : 0.8
COMET INFO:     verbose                        : False
COMET INFO:     world_size                     : 1
COMET INFO:   Uploads:
COMET INFO:     environment details : 1
COMET INFO:     filename            : 1
COMET INFO:     installed packages  : 1
COMET INFO:     model graph         : 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 to ensure all metrics and code
COMET INFO: Experiment is live on comet.com https://www.comet.com/debmalyaray9989/general/a7d823036a8b42a8b4f78395bc808e23

{'web': 'https://www.comet.com/api/asset/download?assetId=88f082613bc6466ab16ebede3b9328e5&experimentKey=a7d823036a8b42a8b4f78395bc808e23',
 'api': 'https://www.comet.com/api/rest/v2/experiment/asset/get-asset?
assetId=88f082613bc6466ab16ebede3b9328e5&experimentKey=a7d823036a8b42a8b4f78395bc808e23',
 'assetId': '88f082613bc6466ab16ebede3b9328e5'}
COMET INFO: Couldn't find a Git repository in '/content' nor in any parent directory. Set `COMET_GIT_DIRECTORY` if your Git Repository is elsewhe
```

Start coding or generate with AI.

Start coding or generate with AI.

Start coding or generate with AI.