#Red Wine Quality Dataset

```
from google.colab import files


uploaded = files.upload()
import pandas as pd
wn = pd.read_csv('winequality2.csv')
wn.head()
wn.isnull().values.any()
wn.describe()

x = wn.drop('quality', axis = 1)
y = wn['quality']
from sklearn.model_selection import train_test_split

from sklearn.pipeline import Pipeline
from sklearn.feature_selection import VarianceThreshold
from imblearn.over_sampling import SMOTE
import numpy as np
from sklearn.datasets import make_classification
rng = np.random.RandomState(42)
x, y = make_classification(random_state=rng)
oversample=SMOTE()
wn=oversample.fit_resample(x,y)
import sklearn.model_selection
!pip install scikit-optimize
from skopt import BayesSearchCV
x_train, x_test, y_train, y_test = sklearn.model_selection.train_test_split(x, y,
random_state=42, stratify=y)
from sklearn.ensemble import RandomForestClassifier
pipe = Pipeline([
    ('selector', VarianceThreshold()),
    ('model', RandomForestClassifier())
])
param_grid = {
    'selector__threshold': (0,0.1),
    'model': [RandomForestClassifier()],
    'model__max_depth': (1,500),
    'model__max_leaf_nodes': (10,100),
    'model__max_samples': (0.1,0.8),
    'model__min_samples_leaf': (1,6),
    'model__min_samples_split': (2,8),
    'model__n_estimators': (50,700),
    'model__n_jobs': (1,400)
}
opt = BayesSearchCV(
    pipe,
    # (parameter space, # of evaluations)
    [(param_grid, 50)],
    cv=3)

from sklearn.model_selection import cross_val_score

scores = cross_val_score(opt, x, y, cv=3)
print(scores)

opt.fit(x_train, y_train)
print("test score: %s" % opt.score(x_test, y_test))
clf = RandomForestClassifier(random_state=42)
clf = clf.fit(x_train, y_train)
y_hat = clf.predict(x_test)
print("RF Accuracy", sklearn.metrics.accuracy_score(y_test, y_hat))
```

# Brain Stroke Dataset

```
from google.colab import files


uploaded = files.upload()
import pandas as pd
st = pd.read_csv('brain_stroke.csv')
st.head()
print(st['gender'].unique())
st['gender'].value_counts()
print(st['ever_married'].unique())
st['ever_married'].value_counts()
print(st['work_type'].unique())
st['work_type'].value_counts()
print(st['Residence_type'].unique())
```

```
st['Residence_type'].value_counts()
print(st['smoking_status'].unique())
st['smoking_status'].value_counts()
ohed = pd.get_dummies(st, columns = ['gender', 'ever_married', 'work_type',
'Residence_type', 'smoking_status'])
ohed.head()
ohed.describe()
from sklearn.model_selection import train_test_split
x = ohed.drop('stroke', axis = 1)
y = ohed['stroke']
import numpy as np
from sklearn.datasets import make_classification
from imblearn.over_sampling import SMOTE
oversample=SMOTE()
ohed=oversample.fit_resample(x,y)
import sklearn.model_selection
x_train, x_test, y_train, y_test = sklearn.model_selection.train_test_split(x, y,
random_state=42, stratify=y)
from sklearn.ensemble import RandomForestClassifier
!pip install scikit-optimize
from skopt import BayesSearchCV
from sklearn.pipeline import Pipeline
from sklearn.feature_selection import VarianceThreshold
x_train, x_test, y_train, y_test = sklearn.model_selection.train_test_split(x, y,
random_state=42, stratify=y)
pipe = Pipeline([
   ('selector', VarianceThreshold()),
   ('model', RandomForestClassifier())
])
rfc_search = {
   'selector__threshold': (0,0.1),
   'model':([RandomForestClassifier()]),
   'model__max_depth': (1,500),
   'model__max_leaf_nodes': (10,100),
   'model__max_samples': (0.1,0.8),
   'model__min_samples_leaf': (1,6),
    'model__min_samples_split': (2,8),
   'model__n_estimators': (50,700),
   'model__n_jobs': (-1,400)
}
```

```
opt = BayesSearchCV(
    pipe,
    # (parameter space, # of evaluations)
    [(rfc_search, 50)],
    cv=3)

from sklearn.model_selection import cross_val_score

scores = cross_val_score(opt, x, y, cv=3)
print(scores)

opt.fit(x_train, y_train)
print("test score: %s" % opt.score(x_test, y_test))
clf = RandomForestClassifier(random_state=42)
clf = clf.fit(x_train, y_train)
y_hat = clf.predict(x_test)
print("RF Accuracy", sklearn.metrics.accuracy_score(y_test, y_hat))
```

Table 1: Red Wine Quality Pipeline	
Mean Accuracy Score Before Bayesian Optimization	0.88
Mean Accuracy Score After Bayesian Optimization	0.88

Table 2: Brain Stroke Pipeline	
Mean Accuracy Score Before Bayesian Optimization	0.9478
Mean Accuracy Score After Bayesian Optimization	0.9502