In [1]:
```
from scipy.io import arff
from io import StringIO
import itertools
import numpy as np
import pandas as pd

from sklearn import tree
from sklearn.ensemble import AdaBoostClassifier, GradientBoostingClassifier, R
andomForestClassifier
from sklearn.naive_bayes import MultinomialNB, BernoulliNB, GaussianNB, Comple
mentNB
from sklearn.neighbors import KNeighborsClassifier
from sklearn.neural_network import MLPClassifier
from sklearn.svm import LinearSVC
from sklearn.linear_model import LogisticRegression

from sklearn.datasets import load_iris, load_breast_cancer

from sklearn.preprocessing import MinMaxScaler, Normalizer, add_dummy_feature,
label_binarize

from sklearn.model_selection import train_test_split, cross_val_score, cross_v
al_predict

from sklearn.metrics import accuracy_score, f1_score
```

In [2]:
```
breast_cancer = load_breast_cancer()
```

In [3]:
```
scl_min_max = MinMaxScaler()
scl_min_max.fit(breast_cancer.data)

scl_norm = Normalizer()
scl_norm.fit(breast_cancer.data)
```

Out[3]:
```
Normalizer(copy=True, norm='l2')
```

Split data into train, validation, testing

In [4]:
```
bc_working_data, bc_validate_data, bc_working_target, bc_validate_target = tra
in_test_split(breast_cancer.data, breast_cancer.target, test_size = 0.1, rando
m_state = 1, shuffle=True, stratify=breast_cancer.target)

bc_data_train, bc_data_test, bc_train_target, bc_test_target = train_test_spli
t(bc_working_data, bc_working_target, test_size = 0.2, random_state = 1,strati
fy=bc_working_target)
```

In [5]:
```
model_names = []
accuracy = []
fscore = []
params = []
```

Decision tree

In [6]:
```
clf_dt = tree.DecisionTreeClassifier()
clf_dt = clf_dt.fit(bc_data_train, bc_train_target)
bc_pred_dt = clf_dt.predict(bc_data_test)

# accuracy of model
model_names.append("Decision Tree")
accuracy.append(accuracy_score(bc_test_target, bc_pred_dt))
fscore.append(f1_score(bc_test_target, bc_pred_dt))
params.append(" ")
```

K-Neighbors

In [7]:
```
clf_kn_5 = KNeighborsClassifier()
clf_kn_5 = clf_kn_5.fit(bc_data_train, bc_train_target)
bc_pred_kn_5 = clf_kn_5.predict(bc_data_test)

# accuracy of model
model_names.append("K-Neighbors")
accuracy.append(accuracy_score(bc_test_target, bc_pred_kn_5))
fscore.append(f1_score(bc_test_target, bc_pred_kn_5))
params.append(" ")
```

Logistic Regression

In [8]:
```
clf_log = LogisticRegression(solver='newton-cg')
clf_log = clf_log.fit(bc_data_train, bc_train_target)
bc_pred_log = clf_log.predict(bc_data_test)

# accuracy of model
model_names.append("Logistic_Regression")
accuracy.append(accuracy_score(bc_test_target, bc_pred_log))
fscore.append(f1_score(bc_test_target, bc_pred_log))
params.append("solver='newton-cg'")
```

Multinomial Naive Bayes

In [9]:
```
clf_nb = MultinomialNB()
clf_nb = clf_nb.fit(bc_data_train, bc_train_target)
bc_pred_nb = clf_nb.predict(bc_data_test)

# accuracy of model
model_names.append("Multinomail_Naive_Bayes")
accuracy.append(accuracy_score(bc_test_target, bc_pred_nb))
fscore.append(f1_score(bc_test_target, bc_pred_nb))
params.append(" ")
```

Random Forest

In [10]:
```
clf_rf = RandomForestClassifier(n_estimators=100)
clf_rf = clf_rf.fit(bc_data_train, bc_train_target)
bc_pred_rf = clf_rf.predict(bc_data_test)

# accuracy of model
model_names.append("Random_Forest")
accuracy.append(accuracy_score(bc_test_target, bc_pred_rf))
fscore.append(f1_score(bc_test_target, bc_pred_rf))
params.append("n_estimators=100")
```

Neural Net

In [11]:
```
bc_data_train_mm = scl_min_max.transform(bc_data_train)
bc_data_test_mm = scl_min_max.transform(bc_data_test)

clf_nn = MLPClassifier(hidden_layer_sizes=(30,30,30,), early_stopping=False, l
earning_rate='adaptive', max_iter=1000)
clf_nn = clf_nn.fit(bc_data_train_mm, bc_train_target)

bc_pred_nn = clf_nn.predict(bc_data_test_mm)

# accuracy of model
model_names.append("Neural_Net")
accuracy.append(accuracy_score(bc_test_target, bc_pred_nn))
fscore.append(f1_score(bc_test_target, bc_pred_nn))
params.append({"hidden_layer_sizes=(30,30,30,)","early_stopping=False","learni
ng_rate='adaptive'","max_iter=1000"})
```

Tabulation of Results

In [12]:
```
df = pd.DataFrame()
df["Model"] = model_names
df["Accuracy"] = accuracy
df["F-score"] = fscore
df["Parameters"] = params
```

In [13]:
```
df
```

Out[13]:

	Model	Accuracy	F-score	Parameters
0	Decision Tree	0.834951	0.866142	
1	K-Neighbors	0.893204	0.917293	
2	Logistic_Regression	0.902913	0.920635	solver='newton-cg'
3	Multinomail_Naive_Bayes	0.815534	0.861314	
4	Random_Forest	0.912621	0.931298	n_estimators=100
5	Neural_Net	0.941748	0.953125	{max_iter=1000, early_stopping=False, learning...