Machine Learning Homework 8

Hairui Yin

1.

(a) After import useful libraries, let the first 150 samples be used for training and rest 50 for testing. Using bootstrap method, we construct the 50 training datasets from as follow:

```python
import pandas as pd
from sklearn.utils import resample

df = pd.read_csv('moonDataset.csv')
train_data = df.iloc[:150, :]
test_data = df.iloc[150:, :]

X_train = train_data.iloc[:, :-1].values
y_train = train_data.iloc[:, -1].values
X_test = test_data.iloc[:, :-1].values
y_test = test_data.iloc[:, -1].values

bootstrap_datasets = []
for i in range(50):
        X_bootstrap, y_bootstrap = resample(X_train, y_train, n_samples=150, random_state=i)
        bootstrap_datasets.append((X_bootstrap, y_bootstrap))
```

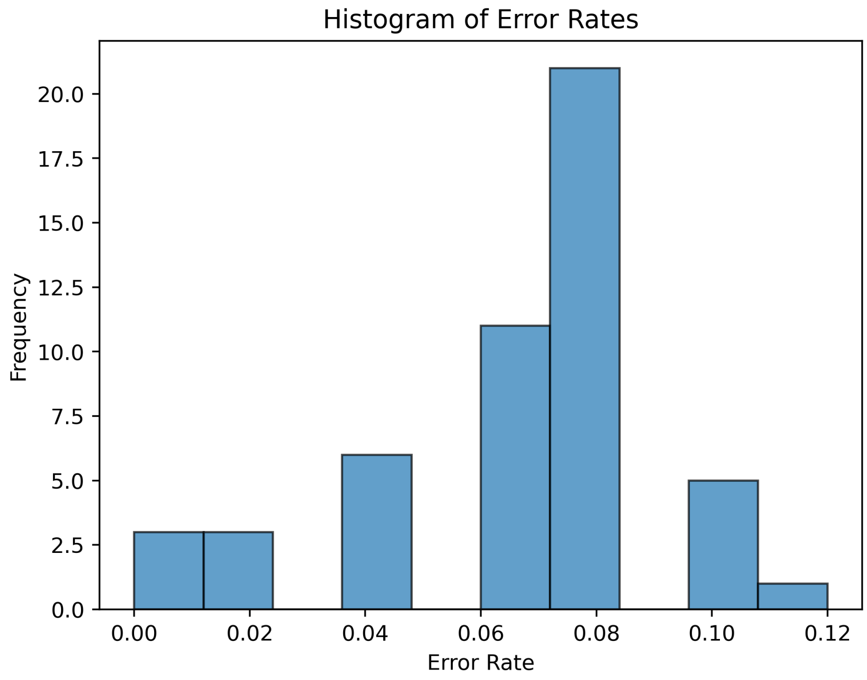
(b) We use MLPClassifier in sklearn lib to construct our network. For each training datasets, we train the model and predict data in test set. Each error rate is recorded and shown in histogram.

```python
from sklearn.neural_network import MLPClassifier
import matplotlib.pyplot as plt
from sklearn.metrics import accuracy_score
```

```python
error_rates = []
for i in range(50):
        # Train a feedforward network
        model = MLPClassifier(hidden_layer_sizes=(10,), max_iter=10000, random_state=i)
        model.fit(bootstrap_datasets[i][0], bootstrap_datasets[i][1])

        # Compute error rate on test dataset
        y_pred = model.predict(X_test)
        error_rate = 1 - accuracy_score(y_test, y_pred)
        error_rates.append(error_rate)
# plot error rate in Histogram
plt.hist(error_rates, bins=10, edgecolor='k', alpha=0.7)
plt.title('Histogram of Error Rates')
plt.xlabel('Error Rate')
plt.ylabel('Frequency')
plt.show()
```

The figure shows below



(c) In this problem, we use decision tree as our classifier. After the code in (a) and (b), we use BaggingClassifier in scikit lib. Then we construct the new binary classifier using bagging with different ensemble size as follows:

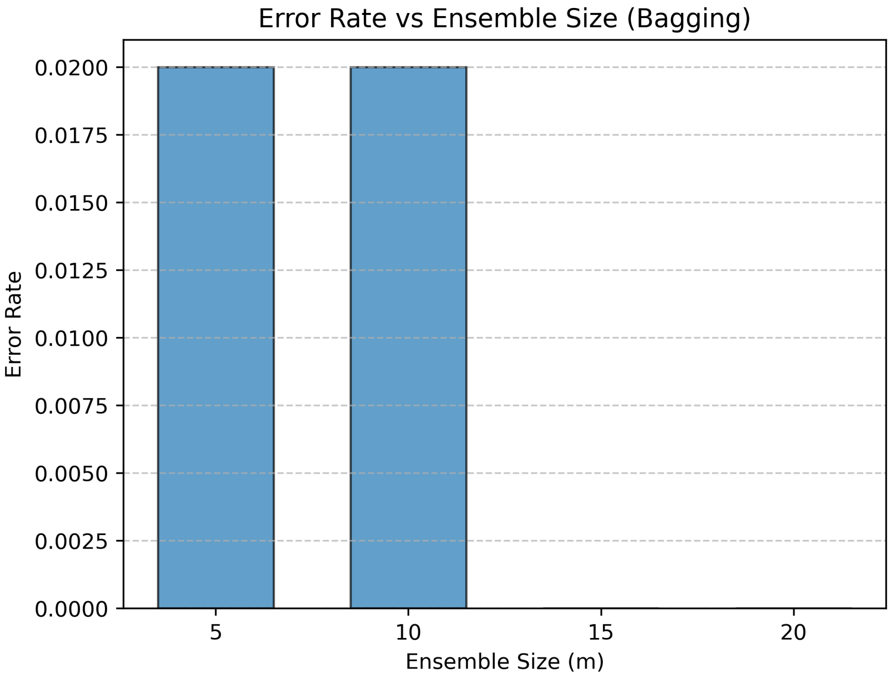
```python
from sklearn.ensemble import BaggingClassifier
from sklearn.tree import DecisionTreeClassifier

ensemble_sizes = [5, 10, 15, 20]
error_rates_bagging = []
for m in ensemble_sizes:
        # Bagging classifier with a decision tree as the base estimator
        bagging_model = BaggingClassifier(
        estimator=DecisionTreeClassifier(),
        n_estimators=m,
        random_state=42
        )
        bagging_model.fit(X_train, y_train)

        # Predict on test data
        y_pred = bagging_model.predict(X_test)

        # Calculate error rate
        error_rate = 1 - accuracy_score(y_test, y_pred)
        error_rates_bagging.append(error_rate)
# Plot
plt.bar(ensemble_sizes, error_rates_bagging, width=3, edgecolor='k', alpha=0.7)
plt.title('Error Rate vs Ensemble Size (Bagging)')
plt.xlabel('Ensemble Size (m)')
plt.ylabel('Error Rate')
plt.xticks(ensemble_sizes)
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.show()
```

The figure shows below


Error Rate vs Ensemble Size (Bagging)
Error Rate
0.0200
0.0175
0.0150
0.0125
0.0100
0.0075
0.0050
0.0025
0.0000
5
10
15
20
Ensemble Size (m)