Genetic Algorithms and Machine Learning Assignment

Chen Shang G2203629G

November 2023

Assignment 1

(1)Bayes decision rule

predicted class labels =
[-1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 1 1 1 1 1 1 1 1 1 1 1]

(2)Naïve Bayes

predicted class labels =
[-1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 1 1 1 1 1 1 1 1 1 1 1]

(3)Linear discriminant analysis

predicted class labels =
[-1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 1 1 1 1 1 1 1 1 1 1 1]

Assignment 2

(1)

Yes. There are some missing values and outliers in the training data.

We have the option to remove rows with missing data, but this approach risks significant data loss, particularly when there's a high degree of missingness.

Another strategy is imputation, where we replace missing values with the mean, median, or mode of the available data. Linear interpolation can also be employed, and it's precisely the method I've chosen. For each missing value, I search for rows with matching features and labels. If found, I replace the missing value with the mean of those similar entries. Otherwise, I resort to linear interpolation.

(2)

Code is in Appendix

(3)

class label of the test data =
[1 1 1 1 1 1 1 1 1 1 3 2 2 2 2 2 2 2 2 2 2 3 3 3 3 3 3 3 3 3]

Appendix

Code of Assignment 1

```
import scipy.io
from sklearn.naive_bayes import GaussianNB
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis as LDA

data_train = scipy.io.loadmat('data_train.mat')['data_train']
data_test = scipy.io.loadmat('data_test.mat')['data_test']
label_train = scipy.io.loadmat('label_train.mat')['label_train']

clf_nb = GaussianNB()
clf_nb.fit(data_train, label_train)
y_pred_nb = clf_nb.predict(data_test)

clf_lda = LDA()
clf_lda.fit(data_train, label_train)
y_pred_lda = clf_lda.predict(data_test)
```

Code of Assignment 2

```
import numpy as np
import pandas as pd
from sklearn.tree import DecisionTreeClassifier

df_train = pd.read_excel('TrainingData.xlsx', header=None, engine='openpyxl')
df_test = pd.read_excel('TestData.xlsx', header=None, engine='openpyxl')
df_train.replace('?', np.nan, inplace=True)
df_train = df_train.astype(float)
df_test = df_test.astype(float)

for index, row in df_train.iterrows():
    if row.isnull().any():
        features = row.iloc[0:4]
        label = row.iloc[4]
        mask1 = (df_train.iloc[:, 0:4] == features).all(axis=1)
        mask2 = df_train.iloc[:, 4] == label
        similar_rows = df_train[mask1 & mask2]

        mean_values = similar_rows.mean()

        df_train.iloc[index] = row.fillna(mean_values)

df_train.interpolate(method='linear', inplace=True)

data_train = df_train.iloc[:, 0:4]
label_train = df_train.iloc[:, 4]
data_test = df_test.iloc[:, 0:4]

clf = DecisionTreeClassifier()
clf.fit(data_train, label_train)
label_pred = clf.predict(data_test).astype(int)
```