Omar Moustafa

900222400

April 30, 2025

CSCE 3601

Assignment 3

1. Considering the given table:

    a. Search for a program to build a classifier for the above table using a decision tree:

The following program is the Python code I found that builds a classifier for the given table using a decision tree:

```
# Importing necessary libraries
import pandas as pd
from sklearn.tree import DecisionTreeClassifier

# Training data
train_data = {
   'Credit_History':
['bad','bad','good','bad','good','good','good','good','bad','bad'],
   'Debt':
['high','low','high','high','low','high','high','low','high','low'],
   'Marital_Status':
['married','single','divorced','divorced','single','married','single','mar
ried','divorced','married'],
   'Income': [30, 25, 35, 45, 48, 55, 70, 38, 70, 90],
   'Accept': ['No', 'No' ,'No' ,'No', 'Yes' ,'Yes' ,'Yes' ,'Yes' ,'No'
,'Yes']
}

df = pd.DataFrame(train_data)

# Convert categorical data to numerical
df_encoded = pd.get_dummies(df[['Credit_History', 'Debt',
'Marital_Status']])
X_train = pd.concat([df_encoded, df['Income']], axis=1)
y_train = df['Accept']
```

```
# Fit decision tree
clf = DecisionTreeClassifier()
clf.fit(X_train, y_train)

# Test applicants
test_data = {
   'CreditHistory': ['good', 'bad', 'bad', 'good'],
   'Debt': ['high', 'low', 'high', 'low'],
   'MaritalStatus': ['married', 'divorced', 'single', 'divorced'],
   'Income': [80, 45, 65, 40]
}
df_test = pd.DataFrame(test_data)

# Encode test data
df_test_encoded = pd.get_dummies(df_test.rename(columns={
   'CreditHistory': 'Credit_History',
   'MaritalStatus': 'Marital_Status'
}))
df_test_encoded = df_test_encoded.reindex(columns=X_train.columns,
fill_value=0)

# Predict
predictions = clf.predict(df_test_encoded)
df_test['AcceptLoan'] = predictions

# Output
print(df_test)
```

b. Given the following records for a set of employees, predict whether to accept loan applications for those employees who have the above data.

The prediction (output to the above code) was the following:

```
  CreditHistory  Debt MaritalStatus  Income AcceptLoan
0          good  high       married      80        Yes
1           bad   low      divorced      45         No
2           bad  high        single      65        Yes
3          good   low      divorced      40         No
```

Comments on the prediction (output):

According to the decision tree, the first and third applications should be approved, while the second and fourth should be turned down. This implies that even in cases where debt or marital status were less favorable, the model gave greater weight to combinations such as high income and good credit history (for Applicant #1) and maybe poor credit history but high income (for Applicant #3).

Despite having a low income and an excellent credit history, the fourth candidate was turned down, perhaps because their "divorced" status did not match previous approved cases. Overall, the decision tree appears to have picked up certain rule-based divides that might prioritize one’s credit history and income level first, rather than one's marital status and debt state.

2. Implement an NB classifier to predict the label of each class row of the table using any programming language you prefer. You can also reuse any public domain code. Then test your classifier using the table given in 1b.

The following program is the Python code I found that builds a classifier for the given table using the Naive Bayes classifier:

```
# Importing necessary libraries
import pandas as pd
from sklearn.naive_bayes import GaussianNB

# Training data
train_data = {
   'CreditHistory':
['bad','bad','good','bad','good','good','good','good','bad','bad'],
   'Debt':
['high','low','high','high','low','high','high','low','high','low'],

'MaritalStatus':['married','single','divorced','divorced','single','marrie
d','single','married','divorced','married'],
   'Income':
[30000,25000,35000,45000,48000,55000,70000,38000,70000,90000],
   'AcceptLoan':
['No','No','No','No','Yes','Yes','Yes','Yes','No','Yes']
}
df = pd.DataFrame(train_data)

# Train model
X = pd.get_dummies(df[['CreditHistory','Debt','MaritalStatus']])
X['Income'] = df['Income']
y = (df['AcceptLoan'] == 'Yes').astype(int)

# New applicants
applications = pd.DataFrame([

{'CreditHistory':'good','Debt':'high','MaritalStatus':'married','Income':
80000},
```

```
   {'CreditHistory':'bad', 'Debt':'low',
'MaritalStatus':'divorced','Income': 45000},
   {'CreditHistory':'bad',
'Debt':'high','MaritalStatus':'single','Income': 65000},
   {'CreditHistory':'good','Debt':'low',
'MaritalStatus':'divorced','Income': 40000},
])

# Encode new applicants
X_new =
pd.get_dummies(applications[['CreditHistory','Debt','MaritalStatus']])
for col in X.columns.drop('Income'):
   if col not in X_new.columns:
       X_new[col] = 0
X_new = X_new[X.columns.drop('Income')]
X_new['Income'] = applications['Income']

# Predict loan approval
model = GaussianNB()
model.fit(X, y)
predictions = model.predict(X_new)

# Output results
applications['AcceptLoan'] = ['Yes' if p == 1 else 'No' for p in
predictions]
print(applications)
```

The prediction (output to the above code) was the following:

```
  CreditHistory  Debt MaritalStatus  Income AcceptLoan
0          good  high       married   80000        Yes
1           bad   low      divorced   45000         No
2           bad  high        single   65000         No
3          good   low      divorced   40000         No
```

Comments on the prediction (output):

Analyzing the above table of employee data and predictions based on Naive Bayes, it can be seen that only Applicant #1 was approved. This person applied while having a good credit

history, high income, and was married, where this combination of characteristics very likely reduced the probability or potential risk of being denied, even though they have a high debt that needs to be paid.

As for the rest of the applications, Applicants #2, #3, and #4 were all predicted to be rejected. Looking at each applicant closely, Applicant #2 was probably denied to be having a bad credit score, even though they also had a low debt that needed to be paid. Application #3 had both a bad credit score along a high debt that they needed to pay, which together must have resulted in their rejection and having very low chances of being approved. Finally, Applicant #4, even though they have both a good credit score and low debt to be paid, their lower income and unideal marital status of being divorced were likely the factors that led to their predicted denial.

Overall, this model using the Naive Bayes' classification shows a major influence of credit history, with bad credit scores tending to lead to being denied. While variables or factors such as one's income or marital status do contribute to the prediction, they seem to be less critical or dominant. This, therefore, highlights the genuine importance of multiple risk factors when it comes to automated decision-making systems.