EE 658/758 Machine Learning in Engineering
Spring 2024

Assignment #3: Neural Network
Due Date: Tuesday, March 19th, 2024

Customer segmentation involves categorizing a company's customer base into distinct groups based on common characteristics such as age, gender, interests, and purchasing behaviors. This strategy is grounded in the understanding that customers are diverse, and marketing is more effective when it is tailored to the specific needs and preferences of smaller, targeted groups. By doing so, companies aim not only to enhance the relevance of their marketing messages, leading to increased sales, but also to gain a deeper insight into what each segment values most. This, in turn, allows for more precise customization of marketing efforts to meet the unique demands of each group.

The data consist of the following columns:

- ID: Unique customer ID (6-digit number)
- Gender: Male, Female
- Ever_Married: Yes, No
- Age: integer value
- Graduated: Yes, No (Is the customer a graduate?)
- Profession: Artist, Healthcare, Entertainment, Engineer, Doctor, Lawyer, Executive, Marketing, Homemaker
- Work_Experience: years working (Numerical value)
- Spending_Score: Low, average, High
- Family_Size: Numerical value (Number of members in the family)
- Var_1: Cat_1, Cat_2, Cat_3, Cat_4, Cat_5, Cat_6, Cat_7 (Anonymized category for the customer)
- Segmentation: A, B, C, D (market segment) - target variable

1. **Data Preprocessing**:
   - Load the dataset and perform initial exploratory data analysis.
   - Handle null values appropriately (e.g., using imputation or removal).
   - Convert categorical variables to a suitable numerical format.
   - Encode the target variable 'Segmentation' for the classification task.
2. **Neural Network Implementation from Scratch:**
   - Implement a two-layer neural network model from scratch focusing on understanding the underlying mathematics and operations (e.g., forward propagation, backpropagation).
   - Apply the model to the preprocessed dataset.
   - Investigate how the choice of activation function Sigmoid vs. ReLU affects the performance
3. **Neural Network using Scikit-learn:**

- Use MLPClassifier class of the scikit-learn library to implement a neural network model with 2 layers.
- Train the model on the preprocessed dataset.
- Repeat with three layers.
- Investigate how the accuracy of the two-layer neural network model varies with the number of neurons in each hidden layer.
- Plot a graph with the number of neurons on the X-axis and the corresponding accuracy on the Y-axis.
- Provide a brief analysis of how the number of neurons in the hidden layers affects the model's performance. Consider discussing any trends, optimal neuron counts, and possible reasons for the observed performance changes.

4. **Logistic Regression Implementation:**
    - Implement a logistic regression model using scikit-learn as a baseline for comparison.
    - Train the model on the same preprocessed dataset.
5. **Evaluation and Comparison:**
    - Display the learning curve for each model to understand the training process.
    - Evaluate the performance of each model using a confusion matrix and classification report.
    - Compare the performance of the neural network models with logistic regression, discussing the strengths and weaknesses of each approach.

Example demonstrating how to check for null values, drop null values, and impute null values in a Pandas DataFrame:

```
import pandas as pd
from sklearn.impute import SimpleImputer

# Sample DataFrame
data = {'Column1': [1, 2, None, 4], 'Column2': [None, 2, 3, 4]}
df = pd.DataFrame(data)

# Checking for null values
print(df.isnull().sum())

# Dropping null values
df_dropped = df.dropna()

# Imputing null values with the mean of the column
imputer = SimpleImputer(strategy='mean')
df_imputed = pd.DataFrame(imputer.fit_transform(df), columns=df.columns)
```

Example on converting a categorical variable into numeric form using one-hot encoding:

```
import pandas as pd

# Sample DataFrame with a categorical column
data = {'Category': ['A', 'B', 'A', 'C']}
df = pd.DataFrame(data)

# Convert the categorical variable into dummy/indicator variables
one_hot_encoded_df = pd.get_dummies(df, columns=['Category'])
print(one_hot_encoded_df)
```

Example on combining two DataFrames side by side:

```
# Sample DataFrames
df1 = pd.DataFrame({'A': ['A0', 'A1', 'A2', 'A3'],
                    'B': ['B0', 'B1', 'B2', 'B3']})


df2 = pd.DataFrame({'C': ['C0', 'C1', 'C2', 'C3'],
                    'D': ['D0', 'D1', 'D2', 'D3']})

# Combine DataFrames side by side
combined_df = pd.concat([df1, df2], axis=1)
print(combined_df)
```