assignment-2

May 18, 2023

1 ASSIGNMENT-2

2 1.Preprocess the dataset

[1]:

```
import pandas as pd
from sklearn.preprocessing import LabelEncoder

# Load the dataset
data = pd.read_csv("House Price India.csv")

# Drop irrelevant features
data = data.drop(["id", "Date"], axis=1)

# Convert categorical features to numerical labels
label_encoder = LabelEncoder()
data["waterfront present"] = label_encoder.fit_transform(data["waterfront␣
  ↪present"])
data["number of bathrooms"] = label_encoder.fit_transform(data["number of␣
  ↪bathrooms"])



# Handle missing values
data = data.dropna()

# Split the dataset into features and target
X = data.drop("Price", axis=1)
y = data["Price"]
```

[5]:

```
from keras.models import Sequential
from keras.layers import Dense

# Initialize the model
model = Sequential()

# Add input layer
model.add(Dense(units=16, activation='relu', input_dim=8))
```

```
# Add at least 2 hidden layers
model.add(Dense(units=8, activation='relu'))
model.add(Dense(units=4, activation='relu'))

# Add output layer
model.add(Dense(units=1, activation='sigmoid'))

# Compile the model
model.compile(loss='binary_crossentropy', optimizer='adam',␣
 ↪metrics=['accuracy'])

# Print model summary
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #   
=================================================================
 dense (Dense)               (None, 16)                144       
                                                                 
 dense_1 (Dense)             (None, 8)                 136       
                                                                 
 dense_2 (Dense)             (None, 4)                 36        
                                                                 
 dense_3 (Dense)             (None, 1)                 5         
                                                                 
=================================================================
Total params: 321
Trainable params: 321
Non-trainable params: 0
_________________________________________________________________
```

[8]:

```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
data = pd.read_csv("House Price India.csv")

# Splitting features and Price variable
X = data.drop('Price', axis=1).values
y = data['Price'].values

# Splitting the dataset into training and testing sets
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
 ↪random_state=42)

# Feature scaling
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)

# Creating the ANN model
model = Sequential()

# Adding the input layer and the first hidden layer
model.add(Dense(units=64, activation='relu', input_dim=X_train.shape[1]))

# Adding additional hidden layers
model.add(Dense(units=64, activation='relu'))
model.add(Dense(units=32, activation='relu'))

# Adding the output layer
model.add(Dense(units=1, activation='sigmoid'))

# Compiling the model
model.compile(optimizer='adam', loss='binary_crossentropy',␣
 ↪metrics=['accuracy'])

# Training the model
model.fit(X_train, y_train, epochs=10, batch_size=32)

# Evaluating the model on the testing set
loss, accuracy = model.evaluate(X_test, y_test)
print("Test loss:", loss)
print("Test accuracy:", accuracy)
```

```
Epoch 1/10
366/366 [==============================] - 1s 2ms/step - loss: -9167181824.0000
- accuracy: 0.0000e+00
Epoch 2/10
366/366 [==============================] - 1s 1ms/step - loss:
-353357856768.0000 - accuracy: 0.0000e+00
Epoch 3/10
366/366 [==============================] - 1s 2ms/step - loss:
-2538013458432.0000 - accuracy: 0.0000e+00
Epoch 4/10
366/366 [==============================] - 1s 2ms/step - loss:
-8876870074368.0000 - accuracy: 0.0000e+00
Epoch 5/10
366/366 [==============================] - 1s 1ms/step - loss:
```

```
-22054615121920.0000 - accuracy: 0.0000e+00
Epoch 6/10
366/366 [==============================] - 1s 1ms/step - loss:
-44977992237056.0000 - accuracy: 0.0000e+00
Epoch 7/10
366/366 [==============================] - 1s 1ms/step - loss:
-80335459057664.0000 - accuracy: 0.0000e+00
Epoch 8/10
366/366 [==============================] - 1s 2ms/step - loss:
-130519459168256.0000 - accuracy: 0.0000e+00
Epoch 9/10
366/366 [==============================] - 1s 2ms/step - loss:
-198271293194240.0000 - accuracy: 0.0000e+00
Epoch 10/10
366/366 [==============================] - 1s 2ms/step - loss:
-285773098123264.0000 - accuracy: 0.0000e+00
92/92 [==============================] - 0s 1ms/step - loss:
-356246263693312.0000 - accuracy: 0.0000e+00
Test loss: -356246263693312.0
Test accuracy: 0.0
```

[12]:

```python
# Assuming you have already trained and saved the model

# Load the preprocessed test dataset
test_data = pd.read_csv("House Price India.csv")

# Splitting features and Price variable
X_test = test_data.drop('Price', axis=1).values
y_test = test_data['Price'].values

# Feature scaling
X_test = scaler.transform(X_test)

# Make predictions on the test set
predictions = model.predict(X_test)

# Convert the probability predictions to binary class labels (0 or 1)
binary_predictions = np.round(predictions).flatten()

# Compare the predictions with the actual labels
accuracy = np.sum(binary_predictions == y_test) / len(y_test)
print("Test accuracy:", accuracy)
```

```
457/457 [==============================] - 1s 1ms/step
Test accuracy: 0.0
```

[ ]: