

Our Solution(s)

Run Code

Your Solutions

Run Code

Solution 1

```
1 // Copyright © 2020 AlgoExpert, LLC. All rights reserved.
2
3 using namespace std;
4
5 class LinkedList {
6 public:
7     int value;
8     LinkedList *next;
9
10    LinkedList(int value) {
11        this->value = value;
12        this->next = NULL;
13    }
14 };
15
16 // O(n) time | O(1) space - where n is the number of nodes in the linked list
17 LinkedList *reverseLinkedList(LinkedList *head) {
18     LinkedList *p1 = NULL;
19     LinkedList *p2 = head;
20     while (p2 != NULL) {
21         LinkedList *p3 = p2->next;
22         p2->next = p1;
23         p1 = p2;
24         p2 = p3;
25     }
26     return p1;
27 }
28
```

Solution 1 Solution 2 Solution 3

```
1 using namespace std;
2
3 class LinkedList {
4 public:
5     int value;
6     LinkedList *next;
7
8     LinkedList(int value) {
9         this->value = value;
10        this->next = NULL;
11    }
12 };
13
14 LinkedList *reverseLinkedList(LinkedList *head) {
15     // Write your code here.
16     return NULL;
17 }
18
```

Our Tests

```
1 // Test Case 1
2 // Input: head = 1->2->3->4->5, expected = 5->4->3->2->1
3 // Output: 5 4 3 2 1
4 // Test Case 2
5 // Input: head = 1->2->3->4->5, expected = 1->2->3->4->5
6 // Output: 1 2 3 4 5
7 // Test Case 3
8 // Input: head = 1->2->3->4->5, expected = 1->2->3->4->5
9 // Output: 1 2 3 4 5
10
```

Custom Output

Submit Code

```
1
2
3
4
5
6
7
8
9
10
```

```
1  # Import necessary libraries
2  import pandas as pd
3  import numpy as np
4  from sklearn.preprocessing import StandardScaler
5  from sklearn.model_selection import train_test_split
6  from sklearn.metrics import mean_squared_error, r2_score
7  from sklearn.linear_model import LinearRegression
8  from sklearn.ensemble import RandomForestRegressor
9  from sklearn.svm import SVR
10 from sklearn.neural_network import MLPRegressor
11
12 # Load the dataset
13 data = pd.read_csv('data.csv')
14
15 # Split the data into features and target variable
16 X = data[['feature1', 'feature2', 'feature3']]
17 y = data['target']
18
19 # Split the data into training and testing sets
20 X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
21
22 # Standardize the features
23 scaler = StandardScaler()
24 X_train = scaler.fit_transform(X_train)
25 X_test = scaler.transform(X_test)
26
27 # Train the models
28 # Linear Regression
29 lr = LinearRegression()
30 lr.fit(X_train, y_train)
31
32 # Random Forest Regressor
33 rf = RandomForestRegressor()
34 rf.fit(X_train, y_train)
35
36 # Support Vector Regression
37 svm = SVR()
38 svm.fit(X_train, y_train)
39
40 # MLP Regressor
41 mlp = MLPRegressor()
42 mlp.fit(X_train, y_train)
43
44 # Evaluate the models
45 # Linear Regression
46 lr_pred = lr.predict(X_test)
47 lr_rmse = np.sqrt(mean_squared_error(y_test, lr_pred))
48 lr_r2 = r2_score(y_test, lr_pred)
49
50 # Random Forest Regressor
51 rf_pred = rf.predict(X_test)
52 rf_rmse = np.sqrt(mean_squared_error(y_test, rf_pred))
53 rf_r2 = r2_score(y_test, rf_pred)
54
55 # Support Vector Regression
56 svm_pred = svm.predict(X_test)
57 svm_rmse = np.sqrt(mean_squared_error(y_test, svm_pred))
58 svm_r2 = r2_score(y_test, svm_pred)
59
60 # MLP Regressor
61 mlp_pred = mlp.predict(X_test)
62 mlp_rmse = np.sqrt(mean_squared_error(y_test, mlp_pred))
63 mlp_r2 = r2_score(y_test, mlp_pred)
```

Run or submit code when you're ready.