```
# Import necessary libraries
import pandas as pd
import numpy as np
from sklearn.impute import SimpleImputer
from sklearn.preprocessing import StandardScaler, MinMaxScaler, OneHotEncoder
from sklearn.ensemble import IsolationForest
```

```
# Load the Titanic dataset
data = pd.read_csv('iris.csv')

# Display the first few rows of the dataset
print(data.head())
```

```
   Id  SepalLengthCm  SepalWidthCm  PetalLengthCm  PetalWidthCm      Species
0   1            5.1           3.5            1.4           0.2  Iris-setosa
1   2            4.9           3.0            1.4           0.2  Iris-setosa
2   3            4.7           3.2            1.3           0.2  Iris-setosa
3   4            4.6           3.1            1.5           0.2  Iris-setosa
4   5            5.0           3.6            1.4           0.2  Iris-setosa
```

```
imputer = SimpleImputer(strategy='mean')

# Perform imputation on numeric columns (excluding 'Id' and 'Species')
data[['SepalLengthCm', 'SepalWidthCm', 'PetalLengthCm', 'PetalWidthCm']] = imputer.fit_transform(
    data[['SepalLengthCm', 'SepalWidthCm', 'PetalLengthCm', 'PetalWidthCm']]
)
data.replace({'Iris-setosa': 0, 'Iris-versicolor': 1, 'Iris-virginica': 2}, inplace=True)

# Check for any missing values after imputation
print("Missing values after imputation:")
print(data.isnull().sum())
```

```
Missing values after imputation:
Id               0
SepalLengthCm    0
SepalWidthCm     0
PetalLengthCm    0
PetalWidthCm     0
Species          0
dtype: int64
```

```
iso_forest = IsolationForest(contamination=0.1, random_state=42)
data['anomaly'] = iso_forest.fit_predict(data[['SepalLengthCm', 'SepalWidthCm', 'PetalLengthCm', 'PetalWidthCm']])

# Identify anomalies
anomalies = data[data['anomaly'] == -1]
print("Detected anomalies:")
print(anomalies)
```

```
Detected anomalies:
      Id  SepalLengthCm  SepalWidthCm  PetalLengthCm  PetalWidthCm  Species  \
13    14            4.3           3.0            1.1           0.1        0
14    15            5.8           4.0            1.2           0.2        0
15    16            5.7           4.4            1.5           0.4        0
22    23            4.6           3.6            1.0           0.2        0
32    33            5.2           4.1            1.5           0.1        0
41    42            4.5           2.3            1.3           0.3        0
60    61            5.0           2.0            3.5           1.0        1
62    63            6.0           2.2            4.0           1.0        1
105  106            7.6           3.0            6.6           2.1        2
109  110            7.2           3.6            6.1           2.5        2
117  118            7.7           3.8            6.7           2.2        2
118  119            7.7           2.6            6.9           2.3        2
122  123            7.7           2.8            6.7           2.0        2
131  132            7.9           3.8            6.4           2.0        2
135  136            7.7           3.0            6.1           2.3        2

     anomaly
13        -1
14        -1
15        -1
22        -1
32        -1
41        -1
60        -1
62        -1
105       -1
109       -1
117       -1
118       -1
122       -1
```

```
131       -1
135       -1
```

```
# Step 4: Standardization
scaler = StandardScaler()

# Standardize the features (excluding 'Id' and the 'anomaly' column)
data[['SepalLengthCm', 'SepalWidthCm', 'PetalLengthCm', 'PetalWidthCm']] = scaler.fit_transform(
    data[['SepalLengthCm', 'SepalWidthCm', 'PetalLengthCm', 'PetalWidthCm']]
)

# Display the standardized data
print("Standardized data:")
print(data.head())
```

```
Standardized data:
   Id  SepalLengthCm  SepalWidthCm  PetalLengthCm  PetalWidthCm  Species  \
0   1      -0.900681      1.032057      -1.341272     -1.312977        0
1   2      -1.143017     -0.124958      -1.341272     -1.312977        0
2   3      -1.385353      0.337848      -1.398138     -1.312977        0
3   4      -1.506521      0.106445      -1.284407     -1.312977        0
4   5      -1.021849      1.263460      -1.341272     -1.312977        0

   anomaly
0        1
1        1
2        1
3        1
4        1
```

```
# Step 5: Normalization
min_max_scaler = MinMaxScaler()

# Normalize the features (excluding 'Id' and the 'anomaly' column)
data[['SepalLengthCm', 'SepalWidthCm', 'PetalLengthCm', 'PetalWidthCm']] = min_max_scaler.fit_transform(
    data[['SepalLengthCm', 'SepalWidthCm', 'PetalLengthCm', 'PetalWidthCm']]
)

# Display the normalized data
print("Normalized data:")
print(data.head())
```

```
Normalized data:
   Id  SepalLengthCm  SepalWidthCm  PetalLengthCm  PetalWidthCm  Species  \
0   1       0.222222      0.625000       0.067797      0.041667        0
1   2       0.166667      0.416667       0.067797      0.041667        0
2   3       0.111111      0.500000       0.050847      0.041667        0
3   4       0.083333      0.458333       0.084746      0.041667        0
4   5       0.194444      0.666667       0.067797      0.041667        0

   anomaly
0        1
1        1
2        1
3        1
4        1
```

```
# One-Hot Encoding for the 'Species' column
data = pd.get_dummies(data, columns=['Species'], drop_first=True)

# Display the dataset after One-Hot Encoding
print("Encoded dataset using One-Hot Encoding:")
print(data.head())
```

```
Encoded dataset using One-Hot Encoding:
   Id  SepalLengthCm  SepalWidthCm  PetalLengthCm  PetalWidthCm  anomaly  \
0   1       0.222222      0.625000       0.067797      0.041667        1
1   2       0.166667      0.416667       0.067797      0.041667        1
2   3       0.111111      0.500000       0.050847      0.041667        1
3   4       0.083333      0.458333       0.084746      0.041667        1
4   5       0.194444      0.666667       0.067797      0.041667        1

   Species_1  Species_2
0      False      False
1      False      False
2      False      False
3      False      False
4      False      False
```

```
data.to_csv('p_iris.csv', index=False)
```