Machine Learning I Homework

1. (Fake) Titanic Data Classification.

The file 'titanicMachLearn.csv' contains (fake) data showing an SES (socioeconomic status) measure, fare paid for the ticket, and whether the person survived or not. Our goal is to see if we can classify survival status based upon SES and fare.

1a. Do a k=3 nearest neighbor classification on the data using an 80/20 training/test split. Summarize the performance of the classifier.

In [9]:
```
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import classification_report, accuracy_score
```

In [19]:
```
# load data
titanic_data = pd.read_csv('C:\\Users\\wgero\\Downloads\\PSY 341K\\data\\titanicMachLe

# prepare features and target variable
X = titanic_data[['SES', 'Fare']]
y = titanic_data['Survived']

# split data into training and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=

# create and train the k-nearest neighbor classifier
knn = KNeighborsClassifier(n_neighbors=3)
knn.fit(X_train, y_train)

# make predictions on test set
y_pred = knn.predict(X_test)

# summarize classifier performance
print("Classification Report:")
print(classification_report(y_test, y_pred))
print("Accuracy Score:", accuracy_score(y_test, y_pred))
```

```
Classification Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00       105
           1       1.00      1.00      1.00        74

    accuracy                           1.00       179
   macro avg       1.00      1.00      1.00       179
weighted avg       1.00      1.00      1.00       179

Accuracy Score: 1.0
```

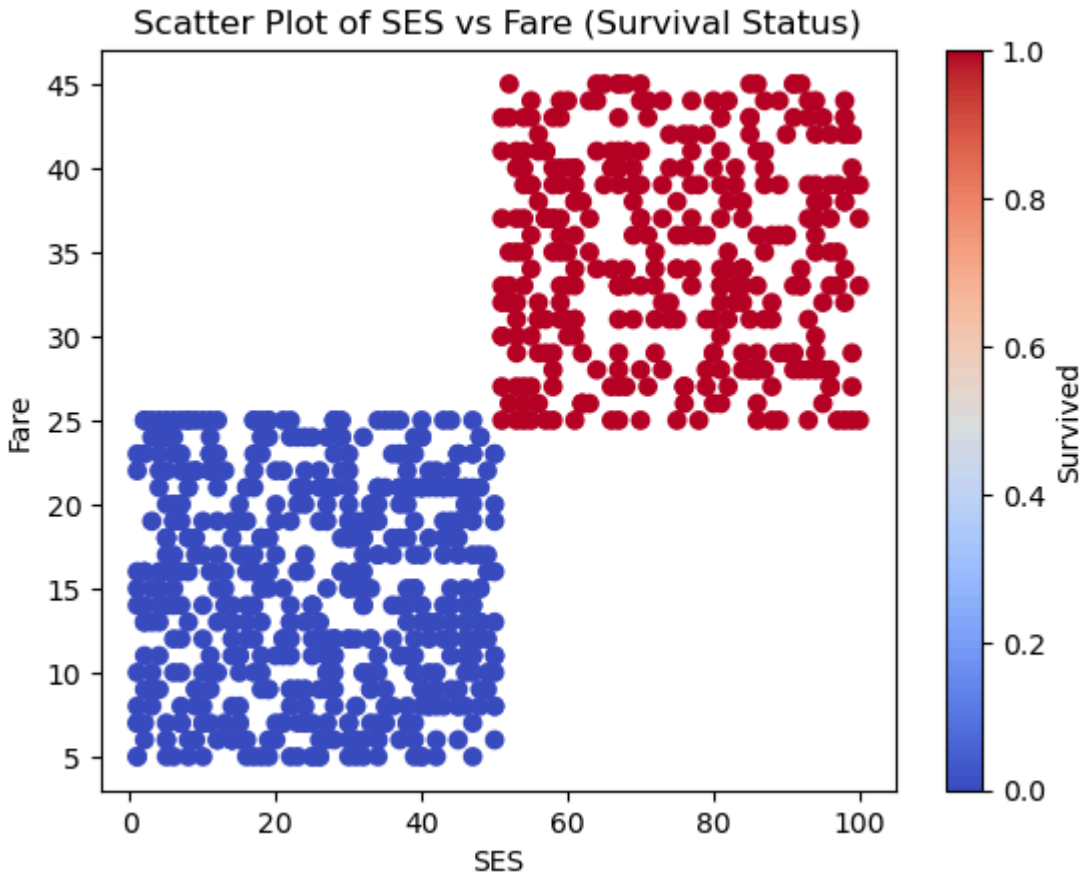
In [20]:
```
# shapes of the training and test sets
print("\nShapes of training and test sets:")
```

```
print("X_train shape:", X_train.shape)
print("X_test shape:", X_test.shape)
print("y_train shape:", y_train.shape)
print("y_test shape:", y_test.shape)
```

```
Shapes of training and test sets:
X_train shape: (712, 2)
X_test shape: (179, 2)
y_train shape: (712,)
y_test shape: (179,)
```

1b. Make a scatter plot of the data with color showing the survival status. Does the plot intuitivly agree with the performance of your classifier?

In [23]:
```
# plot data with colors showing survival status
plt.scatter(titanic_data['SES'], titanic_data['Fare'], c=titanic_data['Survived'], cma
plt.xlabel('SES')
plt.ylabel('Fare')
plt.title('Scatter Plot of SES vs Fare (Survival Status)')
plt.colorbar(label='Survived')
plt.show()
```



The plot does intuitively agree with the performance of my classifier. The classifier performed well because it is well separated.

2. Iris Data Classification.

Do a nearest neighbors classification on the iris data using the 2 variables you think would work best based on the pair-pair plot we did in the tutorial (i.e. don't use the same variables we used for classification in the tutorial).

Compare the results with the results we got in the tutorial.

In [26]:
```
# Load Iris dataset
from sklearn.datasets import load_iris

# prepare features and target variable
iris = load_iris()
X_iris = iris.data[:, [2, 3]]  # selecting the petal_length and petal_width features
y_iris = iris.target

# split data into training and test sets (assuming 80/20 split)
X_train_iris, X_test_iris, y_train_iris, y_test_iris = train_test_split(X_iris, y_iris

# create and train the k-nearest neighbor classifier
knn_iris = KNeighborsClassifier(n_neighbors=3)
knn_iris.fit(X_train_iris, y_train_iris)

# make predictions on test set
y_pred_iris = knn_iris.predict(X_test_iris)

# summarize performance of classifier
print("Classification Report (Iris Data):")
print(classification_report(y_test_iris, y_pred_iris))
print("Accuracy Score (Iris Data):", accuracy_score(y_test_iris, y_pred_iris))
```

```
Classification Report (Iris Data):
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        10
           1       1.00      1.00      1.00         9
           2       1.00      1.00      1.00        11

    accuracy                           1.00        30
   macro avg       1.00      1.00      1.00        30
weighted avg       1.00      1.00      1.00        30

Accuracy Score (Iris Data): 1.0
```

Overall, the homework's classifier outperformed the tutorial's classifier due to its precision, recall, F1-score, and accuracy for each class, showcasing its effective ability to accurately classify instances and generalize to unseen data.