In [2]:
```python
def binning(data, bins):
    min_val = min(data)
    max_val = max(data)
    bin_width = (max_val - min_val) / bins
    binned_data = []

    for value in data:
        bin_index = int((value - min_val) / bin_width)
        if bin_index == bins:
            bin_index -= 1
        binned_data.append(f"Bin-{bin_index}")
    return binned_data

data = [5, 7, 10, 15, 18, 21, 25]
print(binning(data, 3))
```

```
['Bin-0', 'Bin-0', 'Bin-0', 'Bin-1', 'Bin-1', 'Bin-2', 'Bin-2']
```

In [15]:
```python
def min_max_normalize(data, new_min=0, new_max=1):
    old_min = min(data)
    old_max = max(data)
    return [((x - old_min) / (old_max - old_min)) * (new_max - new_min) + new_min f

data = [10, 15, 20, 25, 30]
print(min_max_normalize(data))
```

```
[0.0, 0.25, 0.5, 0.75, 1.0]
```

In [17]:
```python
#Chi Square Test
```

In [19]:
```python
def confusion_matrix(actual, predicted):
    tp = tn = fp = fn = 0
    for a, p in zip(actual, predicted):
        if a == 1 and p == 1:
            tp += 1
        elif a == 0 and p == 0:
            tn += 1
        elif a == 0 and p == 1:
            fp += 1
        elif a == 1 and p == 0:
            fn += 1
    return {"TP": tp, "TN": tn, "FP": fp, "FN": fn}

# Example
actual = [1, 0, 1, 0, 1]
predicted = [1, 0, 0, 0, 1]
print(confusion_matrix(actual,predicted))
```

```
{'TP': 2, 'TN': 2, 'FP': 0, 'FN': 1}
```

In [ ]: