In [1]:

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import classification_report
import matplotlib.pyplot as plt
from sklearn.metrics import accuracy_score
df = pd.read_csv('seattle-weather.csv')
X = df.drop(columns=['date', 'weather'])
Y = df['weather']
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.3, rar
nb_model = GaussianNB()
nb_model.fit(X_train, Y_train)
predictions = nb_model.predict(X_test)
print(predictions)
print(classification_report(Y_test, predictions))
accuracy = accuracy_score(Y_test, predictions) * 100
print(f'Accuracy: {accuracy:.2f}%')
```

```
['sun' 'rain' 'sun' 'sun' 'rain' 'rain' 'sun' 'sun' 'rain' 'sun' 'sun'
 'sun' 'rain' 'sun' 'sun' 'rain' 'sun' 'sun' 'sun' 'sun' 'rain' 'sun'
 'rain' 'sun' 'sun' 'rain' 'sun' 'sun' 'sun' 'sun' 'rain' 'sun' 'sun'
 'rain' 'sun' 'rain' 'rain' 'rain' 'rain' 'sun' 'rain' 'sun' 'sun' 'sun'
 'sun' 'rain' 'rain' 'sun' 'rain' 'sun' 'sun' 'rain' 'sun' 'rain' 'sun'
 'rain' 'rain' 'sun' 'rain' 'sun' 'rain' 'sun' 'rain' 'sun' 'sun' 'sun'
 'rain' 'sun' 'rain' 'rain' 'rain' 'sun' 'sun' 'rain' 'rain' 'sun' 'rain'
 'sun' 'rain' 'snow' 'rain' 'sun' 'rain' 'sun' 'rain' 'sun' 'sun' 'rain'
 'sun' 'rain' 'rain' 'sun' 'sun' 'rain' 'snow' 'sun' 'sun' 'rain' 'sun'
 'rain' 'sun' 'sun' 'rain' 'snow' 'rain' 'sun' 'sun' 'rain' 'sun' 'rain'
 'rain' 'sun' 'rain' 'rain' 'drizzle' 'drizzle' 'sun' 'sun' 'sun' 'rain'
 'rain' 'rain' 'rain' 'rain' 'sun' 'sun' 'sun' 'sun' 'sun' 'sun' 'sun'
 'sun' 'sun' 'sun' 'sun' 'sun' 'rain' 'rain' 'rain' 'sun' 'sun' 'sun'
 'rain' 'rain' 'sun' 'sun' 'sun' 'rain' 'sun' 'sun' 'rain' 'sun' 'sun'
 'sun' 'rain' 'rain' 'sun' 'sun' 'rain' 'rain' 'sun' 'sun' 'rain' 'sun'
 'sun' 'rain' 'rain' 'sun' 'sun' 'sun' 'rain' 'rain' 'sun' 'sun' 'rain'
 'rain' 'sun' 'sun' 'sun' 'rain' 'sun' 'sun' 'sun' 'rain' 'rain' 'sun'
 'rain' 'rain' 'sun' 'sun' 'sun' 'sun' 'sun' 'rain' 'sun' 'sun' 'sun'
 'rain' 'snow' 'rain' 'rain' 'sun' 'sun' 'rain' 'sun' 'rain' 'rain' 'sun'
 'snow' 'sun' 'rain' 'sun' 'sun' 'sun' 'sun' 'sun' 'sun' 'sun' 'rain'
 'sun' 'rain' 'drizzle' 'sun' 'sun' 'sun' 'rain' 'sun' 'rain' 'sun' 'snow'
 'sun' 'rain' 'sun' 'sun' 'rain' 'sun' 'sun' 'sun' 'sun' 'sun' 'rain'
 'rain' 'sun' 'snow' 'sun' 'sun' 'sun' 'sun' 'rain' 'rain' 'rain' 'rain'
 'rain' 'sun' 'sun' 'sun' 'rain' 'sun' 'rain' 'rain' 'rain' 'rain' 'sun'
 'rain' 'sun' 'sun' 'rain' 'sun' 'sun' 'sun' 'sun' 'sun' 'sun' 'rain'
 'sun' 'sun' 'rain' 'rain' 'sun' 'sun' 'sun' 'rain' 'sun' 'sun' 'sun'
 'sun' 'rain' 'sun' 'sun' 'rain' 'sun' 'rain' 'sun' 'sun' 'rain' 'rain'
 'sun' 'rain' 'rain' 'rain' 'rain' 'rain' 'rain' 'rain' 'rain' 'rain'
 'rain' 'sun' 'sun' 'sun' 'rain' 'rain' 'rain' 'sun' 'rain' 'sun' 'sun'
 'rain' 'sun' 'sun' 'rain' 'rain' 'rain' 'sun' 'sun' 'rain' 'snow'
 'drizzle' 'sun' 'sun' 'sun' 'rain' 'rain' 'sun' 'sun' 'sun' 'rain' 'rain'
 'rain' 'rain' 'rain' 'sun' 'rain' 'rain' 'sun' 'rain' 'sun' 'sun' 'rain'
 'rain' 'sun' 'sun' 'sun' 'rain' 'sun' 'rain' 'sun' 'sun' 'sun' 'sun'
 'sun' 'sun' 'sun' 'rain' 'rain' 'rain' 'sun' 'sun' 'sun' 'sun' 'rain'
 'rain' 'rain' 'rain' 'rain' 'sun' 'rain' 'rain' 'rain' 'sun' 'rain' 'sun'
 'sun' 'sun' 'sun' 'rain' 'rain' 'sun' 'drizzle' 'rain' 'rain' 'sun' 'sun'
 'sun' 'rain' 'sun' 'sun' 'sun' 'rain' 'rain' 'rain' 'sun' 'sun' 'sun'
 'sun' 'rain' 'sun' 'sun' 'sun' 'sun' 'sun' 'drizzle' 'sun' 'sun' 'sun'
 'sun' 'rain' 'rain' 'sun' 'sun' 'sun' 'sun' 'sun' 'sun' 'sun' 'rain'
 'sun' 'sun' 'rain' 'sun' 'rain' 'sun' 'sun' 'sun' 'rain' 'rain' 'sun'
 'rain' 'sun']
              precision    recall  f1-score   support

     drizzle       0.33      0.14      0.20        14
         fog       0.00      0.00      0.00        32
        rain       0.98      0.90      0.94       192
        snow       0.50      0.50      0.50         8
         sun       0.77      0.98      0.86       193

    accuracy                           0.84       439
   macro avg       0.52      0.51      0.50       439
weighted avg       0.78      0.84      0.80       439

Accuracy: 84.05%
```

```
C:\PerfLogs\Lib\site-packages\sklearn\metrics\_classification.py:1509: UndefinedMetricWarning: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
C:\PerfLogs\Lib\site-packages\sklearn\metrics\_classification.py:1509: UndefinedMetricWarning: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
C:\PerfLogs\Lib\site-packages\sklearn\metrics\_classification.py:1509: UndefinedMetricWarning: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
```

In [3]:

```python
test_weather={ 'precipitation':0,
'temp_max':20,
'temp_min':10,
'wind':5.4
}
test_df=pd.DataFrame([test_weather])
test_df
```

Out[3]:

	precipitation	temp_max	temp_min	wind
0	0	20	10	5.4

In [5]:

```python
predicted_weather = nb_model.predict(test_df)
print(predicted_weather)
```

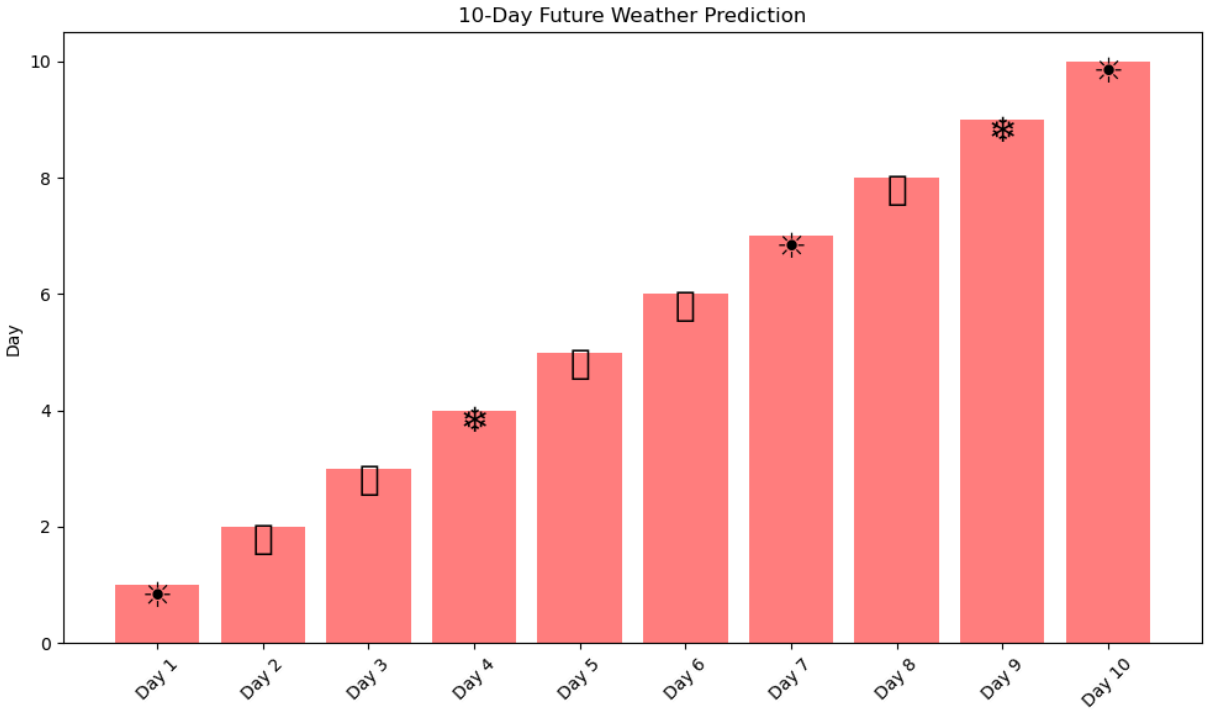
```
['sun']
```

In [63]:

```python
# Sample data for testing
predicted_weather_future = ['sun', 'rain', 'fog', 'snow', 'drizzle', 'rain',

# Dictionary for mapping weather to symbols
weather_symbols = {
    'sun': '☀',
    'rain': '🌧',
    'drizzle': '🌦',
    'fog': '🌫',
    'snow': '❄'
}

# Convert predictions to corresponding symbols
predicted_weather_symbols = [weather_symbols.get(weather, '❓') for weather

# Create days list
days = [f"Day {i+1}" for i in range(10)]

# Plotting
plt.figure(figsize=(10, 6))
bar_heights = range(1, 11)  # Bar heights for days
plt.bar(days, bar_heights, color='red', alpha=0.5)
```

```
# Add weather symbols to bars (positioned at the top of each bar)
for i, symbol in enumerate(predicted_weather_symbols):
    plt.text(i, bar_heights[i] - 0.5, symbol, fontsize=20, ha='center', va='

plt.title("10-Day Future Weather Prediction")
plt.ylabel("Day")
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```

```
C:\Users\19014\AppData\Local\Temp\ipykernel_7752\3317528715.py:31: UserWarni
ng: Glyph 127783 (\N{CLOUD WITH RAIN}) missing from current font.
  plt.tight_layout()
C:\Users\19014\AppData\Local\Temp\ipykernel_7752\3317528715.py:31: UserWarni
ng: Glyph 127787 (\N{FOG}) missing from current font.
  plt.tight_layout()
C:\Users\19014\AppData\Local\Temp\ipykernel_7752\3317528715.py:31: UserWarni
ng: Glyph 127782 (\N{WHITE SUN BEHIND CLOUD WITH RAIN}) missing from current
font.
  plt.tight_layout()
C:\PerfLogs\Lib\site-packages\IPython\core\pylabtools.py:170: UserWarning: G
lyph 127783 (\N{CLOUD WITH RAIN}) missing from current font.
  fig.canvas.print_figure(bytes_io, **kw)
C:\PerfLogs\Lib\site-packages\IPython\core\pylabtools.py:170: UserWarning: G
lyph 127787 (\N{FOG}) missing from current font.
  fig.canvas.print_figure(bytes_io, **kw)
C:\PerfLogs\Lib\site-packages\IPython\core\pylabtools.py:170: UserWarning: G
lyph 127782 (\N{WHITE SUN BEHIND CLOUD WITH RAIN}) missing from current fon
t.
  fig.canvas.print_figure(bytes_io, **kw)
```



In [ ]: