Sentiment Visual Class

```python
## Author : Ashantha Rosary James
import matplotlib.pyplot as plt

class SentimentPlotter:
    def __init__(self, df, data_type='percentage'):
        """
        Initialize the class with a Spark DataFrame.
        :param df: Spark DataFrame containing sentiment data
        :param data_type: Type of data ('percentage' or 'count')
        """
        self.df = df
        self.data_type = data_type

    def extract_data(self):
        """
        Extract sentiment data based on the type of data (percentage or count).
        :return: Two lists, sentiments and percentages or counts
        """
        if self.data_type == 'percentage':
            sentiments = self.df.select('Sentiment').rdd.flatMap(lambda x: x).collect()
            percentages = self.df.select('Percentage').rdd.flatMap(lambda x: x).collect()
            return sentiments, percentages
        elif self.data_type == 'count':
            rdd = self.df.rdd.map(lambda row: (row.Sentiment, row["count"]))
            data = rdd.collect()
            sentiments = [x[0] for x in data]
            counts = [x[1] for x in data]
            return sentiments, counts
```

```
    def plot_data(self, sentiments, values):
        """
        Plot the sentiment distribution as a bar chart (percentage or count).
        :param sentiments: List of sentiment categories
        :param values: List of percentages or counts for each category
        """
        if self.data_type == 'percentage':
            labels = {0: 'Negative', 2: 'Positive'}
            colors = ['limegreen', 'orangered']
            ylabel = 'Percentage'
            title = 'Sentiment Percentage Distribution'
        else:
            labels = {0: 'Negative', 1: 'Neutral', 2: 'Positive'}
            colors = ['pink', 'orange', 'purple']
            ylabel = 'Count'
            title = 'Sentiment Count Distribution'

        plt.figure(figsize=(8, 6))
        bars = plt.bar(sentiments, values, color=colors, tick_label=[labels[s] for s in sentiments])

        for bar in bars:
            yval = bar.get_height()
            plt.text(bar.get_x() + bar.get_width() / 2, yval + (1 if self.data_type == 'percentage' else 10),
                     f'{yval:.2f}%' if self.data_type == 'percentage' else int(yval), ha='center', va='bottom')
        plt.xlabel('Sentiment')
        plt.ylabel(ylabel)
        plt.title(title)
        plt.show()
```