```
class Library:
    def __init__(self):
        self.books = []

    def add_book(self, title, cost):
        self.books.append({'title': title, 'cost': cost})

    def delete_duplicates(self):
        seen_titles = set()
        unique_books = []
        for book in self.books:
            if book['title'] not in seen_titles:
                unique_books.append(book)
                seen_titles.add(book['title'])
        self.books = unique_books

    def display_books_by_cost(self):
        sorted_books = sorted(self.books, key=lambda x: x['cost'])
        for book in sorted_books:
            print(f"Title: {book['title']}, Cost: {book['cost']}")

    def count_books_costing_more_than_500(self):
        count = sum(1 for book in self.books if book['cost'] > 500)
        return count

    def copy_books_costing_less_than_500(self):
        new_list = [book for book in self.books if book['cost'] < 500]
        return new_list

# Example usage:
library = Library()
library.add_book("Book A", 300)
library.add_book("Book B", 600)
library.add_book("Book C", 200)
library.add_book("Book A", 300)
library.add_book("Book D", 800)

print("Books after removing duplicates:")
library.delete_duplicates()
for book in library.books:
    print(f"Title: {book['title']}, Cost: {book['cost']}")

print("\nBooks sorted by cost:")
library.display_books_by_cost()

count = library.count_books_costing_more_than_500()
print(f"\nNumber of books with cost more than 500: {count}")

new_list = library.copy_books_costing_less_than_500()
print("\nBooks with cost less than 500:")
for book in new_list:
    print(f"Title: {book['title']}, Cost: {book['cost']}")
```

Explanation

1)delete_duplicates Method:

This method removes duplicate books based on the book title.

It uses a set to keep track of seen titles and creates a new list with unique books.

2) display_books_by_cost Method:

This method sorts the books by cost in ascending order.

It uses the sorted function with a lambda function as the key to sort by cost.

3) count_books_costing_more_than_500 Method:

This method counts the number of books that cost more than 500.
It uses a generator expression within the sum function to count these books.

4) copy_books_costing_less_than_500 Method:

This method creates a new list of books that cost less than 500.
It uses a list comprehension to filter the books.
The example usage demonstrates how to add books to the library, remove duplicates, display books sorted by cost, count books costing more than 500, and copy books costing less than 500.