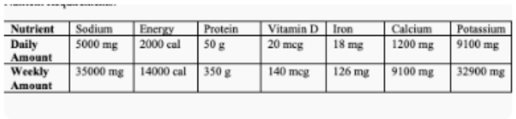


Nutrient	Sodium	Energy	Protein	Vitamin D	Iron	Calcium	Potassium
Daily Amount	5000 mg	2000 cal	50 g	20 mcg	18 mg	1200 mg	9100 mg
Weekly Amount	35000 mg	14000 cal	350 g	140 mcg	126 mg	9100 mg	32900 mg

Nutrient Requirements:

Nutrient	Sodium	Energy	Protein	Vitamin D	Iron	Calcium	Potassium
Daily Amount	5000 mg	2000 cal	50 g	20 mcg	18 mg	1200 mg	9100 mg
Weekly Amount	35000 mg	14000 cal	350 g	140 mcg	126 mg	9100 mg	32900 mg

Appendix B

Food Items and Nutrients:

Item	Sodium (mg)	Energy (cal)	Protein (g)	Vitamin D (mcg)	Iron (mg)	Calcium (mg)	Potassium (mg)	Cost per serving (dollars)
Overnight Oats	210	270	12	0	1.5	30	290	1.99
Mango Chili Salad	240	130	3	0	1.2	50	280	1.33
Cottage Cheese	320	110	12	0.2	0	100	100	0.50
Tofu	15	130	14	0	2.7	60	110	1.35
Chicken Bowl	630	370	22	0	2.6	130	690	2.69
Salmon Burger	330	100	15	20.9	0.3	0	320	1.8725

I can help you work with the data in the image. It seems to be a table showing nutrient requirements and a list of food items with their respective nutrient content. Here's what I can do with it:

1. **Calculate nutrient intake:** I can help you determine how much of each nutrient you're consuming by selecting specific items or creating meal plans.
2. **Optimize cost per meal:** If you're looking to minimize cost while meeting nutritional needs, we can apply optimization techniques, such as linear programming, to find the best combinations.
3. **Compare daily or weekly values:** I can help compare food item nutrients to daily or weekly nutrient requirements to ensure dietary goals are met.

How would you like to proceed with this data?

I have these requirements. I want to create a model to minimize the cost of weekly groceries. Please describe how to do this in english and then provide python code to complete this task

Approach to Minimize Weekly Grocery Cost

You want to create a model to minimize the cost of weekly groceries while meeting the weekly nutritional requirements provided. This is a classic **Linear Programming (LP)** problem where you aim to minimize a cost function subject to constraints on nutrient intake.

Here's how to approach the problem step by step:

1. Define Decision Variables:

Let the decision variables represent the number of servings of each food item you'll consume per week. For example:

- `x1` : Number of servings of Overnight Oats per week
- `x2` : Number of servings of Mango Chili Salad per week
- `x3` : Number of servings of Cottage Cheese per week
- `x4` : Number of servings of Tofu per week
- `x5` : Number of servings of Chicken Bowl per week
- `x6` : Number of servings of Salmon Burger per week

These will be continuous variables, constrained to be non-negative.

2. Objective Function:

The goal is to minimize the total cost of food for the week. The objective function would be:

$$\text{Minimize Cost} = 1.99x1 + 1.33x2 + 0.50x3 + 1.35x4 + 2.69x5 + 1.8725x6$$

3. Constraints:

The constraints ensure that the food chosen meets or exceeds the weekly nutrient requirements. The weekly nutrient constraints are based on the table of food items and nutrients.

For each nutrient:

- **Sodium**: Total sodium intake from the selected servings should be at least 35,000 mg.
- **Energy**: Total energy (calories) intake should be at least 14,000 calories.
- **Protein**: Total protein intake should be at least 350 grams.
- **Vitamin D**: Total vitamin D intake should be at least 140 mcg.
- **Iron**: Total iron intake should be at least 126 mg.
- **Calcium**: Total calcium intake should be at least 9100 mg.
- **Potassium**: Total potassium intake should be at least 32,900 mg.

The constraints for each nutrient would look like:

- Sodium: $210x1 + 240x2 + 320x3 +$ [illegible]$x4 + 630x5 + 330x6 \geq 35,000$

- Sodium: $210x1 + 240x2 + 320x3 + 15x4 + 630x5 + 330x6 \geq 35,000$
- Energy: $270x1 + 130x2 + 110x3 + 130x4 + 370x5 + 100x6 \geq 14,000$
- Protein: $12x1 + 3x2 + 12x3 + 14x4 + 22x5 + 15x6 \geq 350$
- Vitamin D: $0x1 + 0x2 + 0.2x3 + 0x4 + 0x5 + 20.9x6 \geq 140$
- Iron: $1.5x1 + 1.2x2 + 0x3 + 2.7x4 + 2.6x5 + 0.3x6 \geq 126$
- Calcium: $30x1 + 50x2 + 100x3 + 60x4 + 130x5 + 0x6 \geq 9100$
- Potassium: $290x1 + 280x2 + 100x3 + 110x4 + 690x5 + 320x6 \geq 32,900$

4. Non-Negativity Constraints:

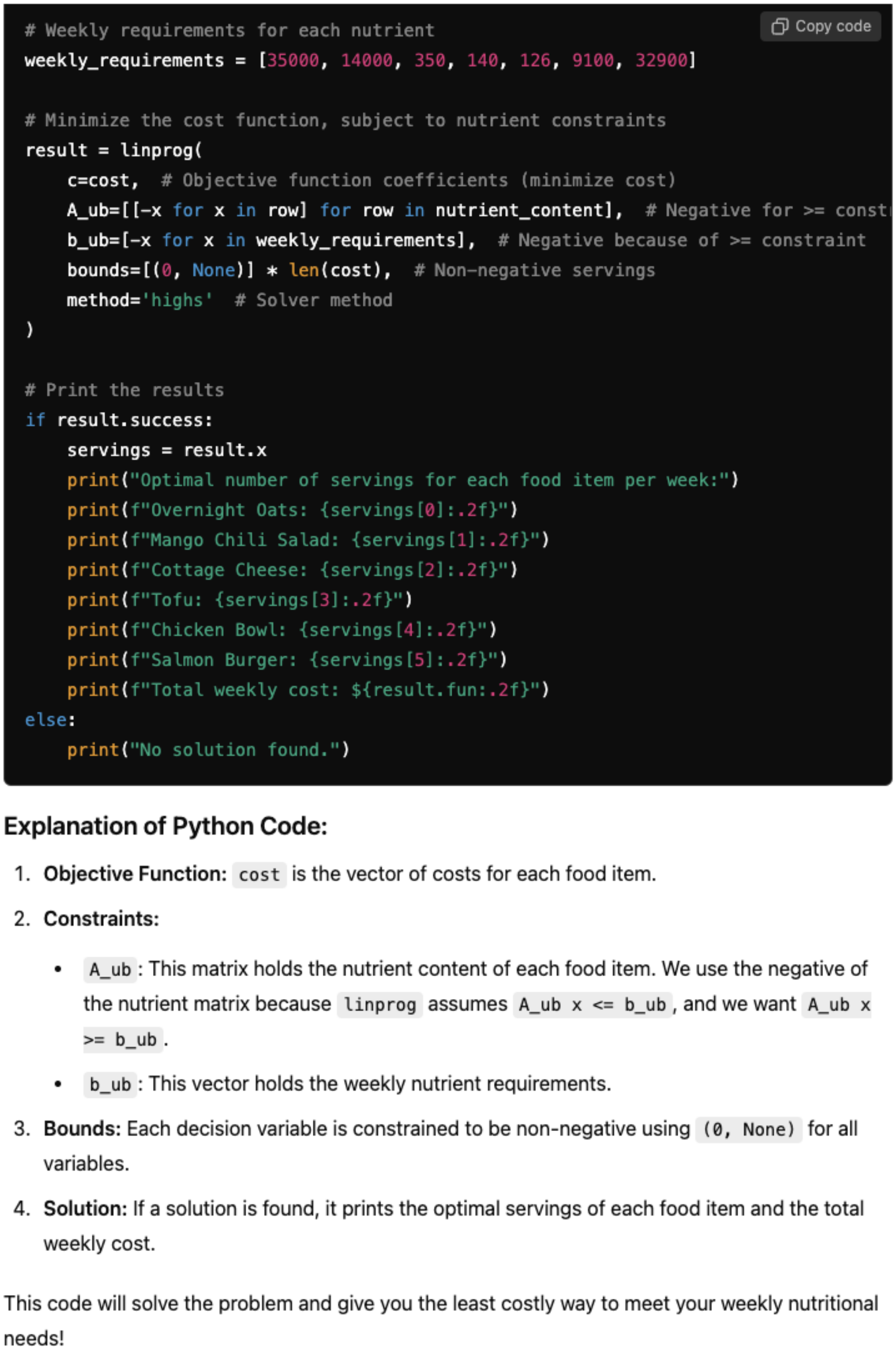
Each decision variable (number of servings) must be non-negative: $x1, x2, x3, x4, x5, x6 \geq 0$

Python Code

We'll use the `scipy.optimize.linprog` function from the `SciPy` library to solve this problem.

python Copy code

```python
from scipy.optimize import linprog

# Cost per serving for each food item
cost = [1.99, 1.33, 0.50, 1.35, 2.69, 1.8725]

# Nutrient content per serving of each food item (rows represent sodium, energy, e
nutrient_content = [
    [210, 240, 320, 15, 630, 330],  # Sodium (mg)
    [270, 130, 110, 130, 370, 100],  # Energy (cal)
    [12, 3, 12, 14, 22, 15],         # Protein (g)
    [0, 0, 0.2, 0, 0, 20.9],         # Vitamin D (mcg)
    [1.5, 1.2, 0, 2.7, 2.6, 0.3],    # Iron (mg)
    [30, 50, 100, 60, 130, 0],       # Calcium (mg)
    [290, 280, 100, 110, 690, 320]   # Potassium (mg)
]

# Weekly requirements for each nutrient
weekly_requirements = [35000, 14000, 350, 140, 126, 9100, 32900]

# Minimize the cost function, subject to nutrient constraints
result = linprog(
    c=cost,  # Objective function coefficients (minimize cost)
    A_ub=[[-x for x in row] for row in nutrient_content],  # Negative for >= const
    b_ub=[-x for x in weekly_requirements],  # Negative because of >= constraint
    bounds=[(0, None)] * len(cost),  # Non-negative servings
    method='highs'  # Solver method
)
```

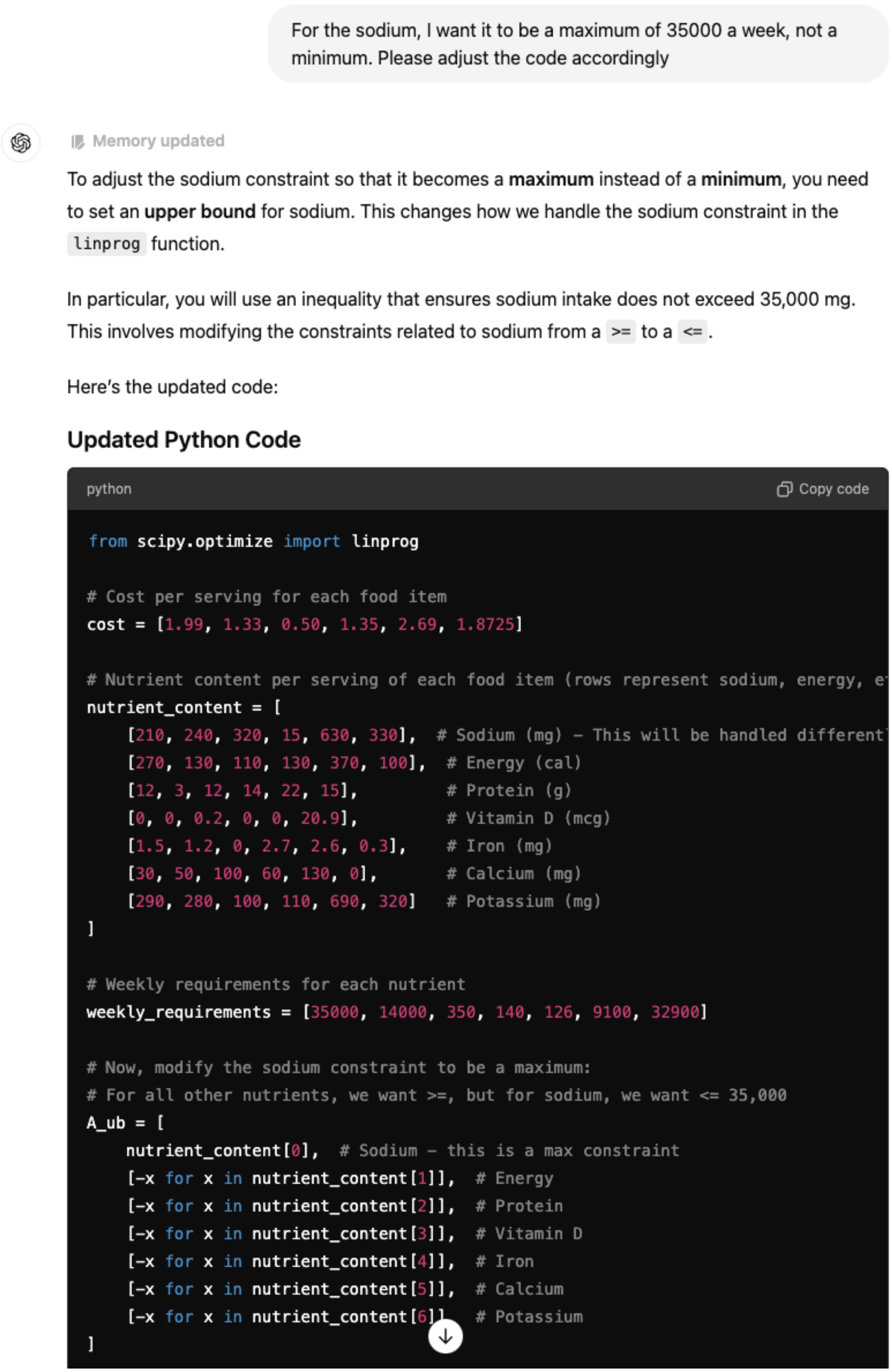


```python
# Weekly requirements for each nutrient
weekly_requirements = [35000, 14000, 350, 140, 126, 9100, 32900]

# Minimize the cost function, subject to nutrient constraints
result = linprog(
    c=cost,  # Objective function coefficients (minimize cost)
    A_ub=[[-x for x in row] for row in nutrient_content],  # Negative for >= const
    b_ub=[-x for x in weekly_requirements],  # Negative because of >= constraint
    bounds=[(0, None)] * len(cost),  # Non-negative servings
    method='highs'  # Solver method
)

# Print the results
if result.success:
    servings = result.x
    print("Optimal number of servings for each food item per week:")
    print(f"Overnight Oats: {servings[0]:.2f}")
    print(f"Mango Chili Salad: {servings[1]:.2f}")
    print(f"Cottage Cheese: {servings[2]:.2f}")
    print(f"Tofu: {servings[3]:.2f}")
    print(f"Chicken Bowl: {servings[4]:.2f}")
    print(f"Salmon Burger: {servings[5]:.2f}")
    print(f"Total weekly cost: ${result.fun:.2f}")
else:
    print("No solution found.")
```

Explanation of Python Code:

1. **Objective Function:** `cost` is the vector of costs for each food item.
2. **Constraints:**
    - `A_ub`: This matrix holds the nutrient content of each food item. We use the negative of the nutrient matrix because `linprog` assumes `A_ub x <= b_ub`, and we want `A_ub x >= b_ub`.
    - `b_ub`: This vector holds the weekly nutrient requirements.
3. **Bounds:** Each decision variable is constrained to be non-negative using `(0, None)` for all variables.
4. **Solution:** If a solution is found, it prints the optimal servings of each food item and the total weekly cost.

This code will solve the problem and give you the least costly way to meet your weekly nutritional needs!



For the sodium, I want it to be a maximum of 35000 a week, not a minimum. Please adjust the code accordingly

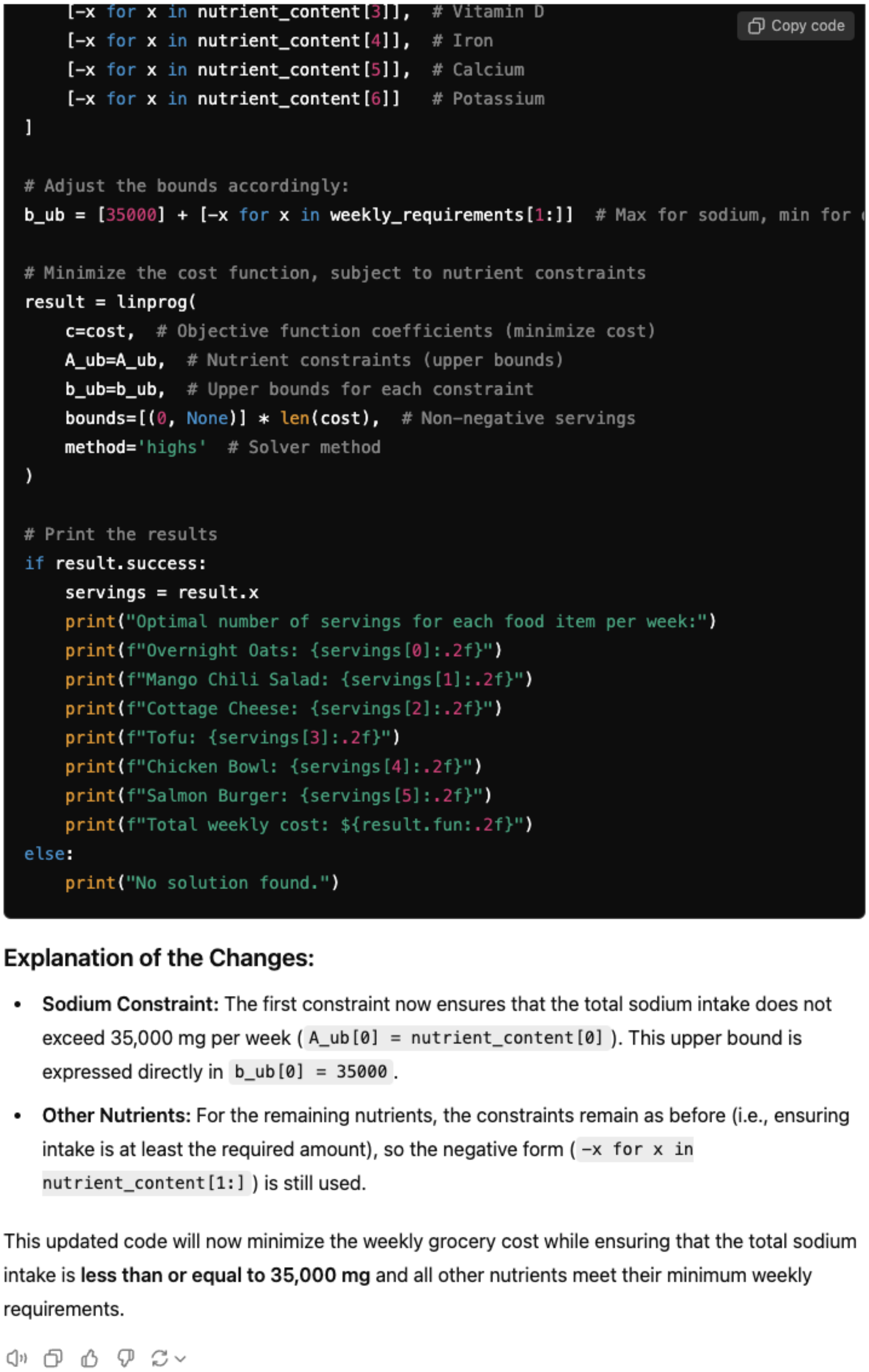
To adjust the sodium constraint so that it becomes a **maximum** instead of a **minimum**, you need to set an **upper bound** for sodium. This changes how we handle the sodium constraint in the `linprog` function.

In particular, you will use an inequality that ensures sodium intake does not exceed 35,000 mg. This involves modifying the constraints related to sodium from a `>=` to a `<=`.

Here's the updated code:

Updated Python Code

```python
from scipy.optimize import linprog

# Cost per serving for each food item
cost = [1.99, 1.33, 0.50, 1.35, 2.69, 1.8725]

# Nutrient content per serving of each food item (rows represent sodium, energy, e
nutrient_content = [
    [210, 240, 320, 15, 630, 330],  # Sodium (mg) - This will be handled different
    [270, 130, 110, 130, 370, 100],  # Energy (cal)
    [12, 3, 12, 14, 22, 15],         # Protein (g)
    [0, 0, 0.2, 0, 0, 20.9],         # Vitamin D (mcg)
    [1.5, 1.2, 0, 2.7, 2.6, 0.3],    # Iron (mg)
    [30, 50, 100, 60, 130, 0],       # Calcium (mg)
    [290, 280, 100, 110, 690, 320]   # Potassium (mg)
]

# Weekly requirements for each nutrient
weekly_requirements = [35000, 14000, 350, 140, 126, 9100, 32900]

# Now, modify the sodium constraint to be a maximum:
# For all other nutrients, we want >=, but for sodium, we want <= 35,000
A_ub = [
    nutrient_content[0],  # Sodium - this is a max constraint
    [-x for x in nutrient_content[1]],  # Energy
    [-x for x in nutrient_content[2]],  # Protein
    [-x for x in nutrient_content[3]],  # Vitamin D
    [-x for x in nutrient_content[4]],  # Iron
    [-x for x in nutrient_content[5]],  # Calcium
    [-x for x in nutrient_content[6]]   # Potassium
]
```



```
    [-x for x in nutrient_content[3]],  # Vitamin D
    [-x for x in nutrient_content[4]],  # Iron
    [-x for x in nutrient_content[5]],  # Calcium
    [-x for x in nutrient_content[6]]   # Potassium
]

# Adjust the bounds accordingly:
b_ub = [35000] + [-x for x in weekly_requirements[1:]]  # Max for sodium, min for

# Minimize the cost function, subject to nutrient constraints
result = linprog(
    c=cost,  # Objective function coefficients (minimize cost)
    A_ub=A_ub,  # Nutrient constraints (upper bounds)
    b_ub=b_ub,  # Upper bounds for each constraint
    bounds=[(0, None)] * len(cost),  # Non-negative servings
    method='highs'  # Solver method
)

# Print the results
if result.success:
    servings = result.x
    print("Optimal number of servings for each food item per week:")
    print(f"Overnight Oats: {servings[0]:.2f}")
    print(f"Mango Chili Salad: {servings[1]:.2f}")
    print(f"Cottage Cheese: {servings[2]:.2f}")
    print(f"Tofu: {servings[3]:.2f}")
    print(f"Chicken Bowl: {servings[4]:.2f}")
    print(f"Salmon Burger: {servings[5]:.2f}")
    print(f"Total weekly cost: ${result.fun:.2f}")
else:
    print("No solution found.")
```

Explanation of the Changes:

- **Sodium Constraint:** The first constraint now ensures that the total sodium intake does not exceed 35,000 mg per week (`A_ub[0] = nutrient_content[0]`). This upper bound is expressed directly in `b_ub[0] = 35000`.
- **Other Nutrients:** For the remaining nutrients, the constraints remain as before (i.e., ensuring intake is at least the required amount), so the negative form (`-x for x in nutrient_content[1:]`) is still used.

This updated code will now minimize the weekly grocery cost while ensuring that the total sodium intake is **less than or equal to 35,000 mg** and all other nutrients meet their minimum weekly requirements.