Advanced Forest Planning

Dr. Bettinger

Chaz Merritt

08/30/2024

Assignment 1- Linear Programming

I implemented the Linear Programming problem using a Python library for modelling LP's called PuLP. The python file is attached, to run it yourself, assuming python is installed on your system, you can install the two dependencies, PuLP and Pandas, like this:

python -m pip install pulp

python -m pip install pandas

According to my formulation I achieved the optimal solution with a total objective value of 0, meaning the deviation from target was completely minimized for each period.

(A) <u>My objective function was:</u>

prob += lpSum(deviation_variables)

This simply means the sum of the "deviation variables" which were calculated as

```
for period in range(1, 9):
harvested_volume = lpSum([
variables[f"stand_{stand}period_{period}"] *
stands_volume.loc[stands_volume['Stand'] == stand, f'P{period}_volume'].values[0]
for stand in range(1, 115)
])

deviation = LpVariable(f"deviation_period_{period}", lowBound=0)

# Deviation constraints (absolute deviation)
prob += deviation >= harvested_volume - target
prob += deviation >= target - harvested_volume

deviation_variables.append(deviation)
```

To enforce the stand adjacency constraint, a binary set of decision variables was created in addition to the standard decision variables which are continuous from 0 to 40. The binary variables would allow me to enforce mutual exclusivity between adjacent stands. Then the rest of the constraints were calculated this way. Note that the age constraint adjusts for the change in age across periods using a 5-year increment. <u>I did not have it written down if we used 5 years, 10 years, or something else between time periods.</u>

(B) Constraints

```
# Can't harvest stands that are less than 35 years old based on stand data (stands age 5 years per period)
for index, row in stands_volume.iterrows():
stand = int(row['Stand'])
age = row['Age']
for period in range(1, 9):
# Calculate the age of the stand at the beginning of the period
period_age = age + (period - 1) * 5
if period_age < 35:
prob += variables[f"stand_{stand}period_{period}"] == 0
# Stand starts at less than 35 years old but becomes mature by the end of the time horizon
elif period_age >= 35:
prob += variables[f"stand_{stand}period_{period}"] <= 40

# Enforce mutual exclusivity for adjacent stands
for index, row in adjacency.iterrows():
stand1 = row['stand']
stand2 = row['adjacent_stand']
for period in range(1, 9):
prob += binary_variables[f"y_stand_{stand1}period_{period}"] +
binary_variables[f"y_stand_{stand2}period_{period}"] <= 1

# Can only harvest each stand once across the entire time horizon
for stand in range(1, 115):
prob += lpSum([binary_variables[f"y_stand_{stand}period_{period}"] for period in range(1, 9)]) <= 1
```

(C) Accounting Rows

I did not use Lindo systems, so the concept of "accounting rows" does not apply directly to my python implementation I don't believe. However, the problem is identical. I calculated my deviations using the absolute difference instead of the square simply because it should be the same and I was having issues with maintaining linearity in PuLP. At any rate, this is how I calculated my deviation variables:

```
deviation_variables = []

for period in range(1, 9):
harvested_volume = lpSum([
variables[f"stand_{stand}period_{period}"] *
stands_volume.loc[stands_volume['Stand'] == stand, f'P{period}_volume'].values[0]
for stand in range(1, 115)
])

deviation = LpVariable(f"deviation_period_{period}", lowBound=0)
```

```
# Deviation constraints (absolute deviation)
prob += deviation >= harvested_volume - target
prob += deviation >= target - harvested_volume

deviation_variables.append(deviation)
```

(D) Solution

Lastly, a solution to the problem is presented on the last pages in the form of my abridged final output. This prints the values for the decision variables after running the solver, which converged to the optimal solution in a tenth of a second. Feel free to check the work and see that every stand for a given period does in fact add up to the target value of 11721.

Here is my abridged final output:

Optimal solution found.
stand_2period_1: 532.0 board feet
stand_10period_1: 600.0 board feet
stand_13period_1: 612.0 board feet
stand_14period_1: 848.0 board feet
stand_19period_1: 1116.0 board feet
stand_37period_1: 380.0 board feet
stand_39period_1: 680.0 board feet
stand_54period_1: 413.0000056 board feet
stand_62period_1: 644.0 board feet
stand_69period_1: 1156.0 board feet
stand_81period_1: 672.0 board feet
stand_84period_1: 1224.0 board feet
stand_104period_1: 1112.0 board feet
stand_107period_1: 856.0 board feet
stand_111period_1: 876.0 board feet
stand_28period_2: 344.0 board feet
stand_38period_2: 680.0 board feet
stand_44period_2: 1180.0 board feet
stand_49period_2: 1140.0 board feet
stand_52period_2: 620.0 board feet
stand_67period_2: 996.0 board feet
stand_68period_2: 553.0000025999999 board feet
stand_72period_2: 1176.0 board feet
stand_89period_2: 1032.0 board feet
stand_92period_2: 1332.0 board feet
stand_95period_2: 1160.0 board feet
stand_99period_2: 436.0 board feet
stand_100period_2: 1072.0 board feet
stand_11period_3: 1480.0 board feet
stand_33period_3: 1200.0 board feet
stand_42period_3: 1288.0 board feet
stand_43period_3: 1328.0 board feet
stand_45period_3: 1012.0 board feet
stand_51period_3: 1140.0 board feet
stand_70period_3: 1328.999994 board feet
stand_74period_3: 1528.0 board feet
stand_112period_3: 1416.0 board feet
stand_22period_4: 2172.0 board feet
stand_25period_4: 1700.9999952 board feet
stand_48period_4: 2168.0 board feet
stand_55period_4: 1348.0 board feet
stand_73period_4: 860.0 board feet
stand_102period_4: 1372.0 board feet
stand_109period_4: 2100.0 board feet
stand_16period_5: 436.0 board feet
stand_24period_5: 737.0000054 board feet
stand_29period_5: 428.0 board feet
stand_31period_5: 972.0 board feet
stand_46period_5: 1968.0 board feet
stand_50period_5: 1740.0 board feet
stand_60period_5: 2376.0 board feet
stand_75period_5: 1792.0 board feet
stand_91period_5: 464.0 board feet
stand_94period_5: 808.0 board feet
stand_4period_6: 2376.0 board feet
stand_6period_6: 680.0 board feet
stand_8period_6: 860.0 board feet
stand_26period_6: 348.0 board feet
stand_56period_6: 2480.0 board feet
stand_63period_6: 712.0 board feet
stand_71period_6: 1136.0 board feet
stand_82period_6: 920.0 board feet
stand_93period_6: 604.0 board feet
stand_96period_6: 464.99999900000006 board feet
stand_105period_6: 520.0 board feet
stand_110period_6: 620.0 board feet
stand_17period_7: 876.0 board feet
stand_27period_7: 1512.0 board feet

stand_35period_7: 884.0
board feet
stand_65period_7: 1460.0
board feet
stand_76period_7: 1960.0
board feet
stand_80period_7:
616.9999988000001 board
feet
stand_87period_7: 1536.0
board feet
stand_108period_7: 2876.0
board feet
stand_12period_8: 1072.0
board feet
stand_20period_8: 1224.0
board feet
stand_41period_8:
1548.9999839999998
board feet
stand_78period_8: 1500.0
board feet
stand_90period_8: 1268.0
board feet
stand_106period_8: 2188.0
board feet
stand_113period_8: 2920.0
board feet