WDWR
Sprawozdanie z projektu
Jan Mogielski 303899
08.04.2024

1. Analityczne sformułowanie modelu

Zadanie dotyczy przedsiębiorstwa produkującego 4 produkty: P1, P2, P3, P4.
Przedsiębiorstwo posiada maszyny, na których wytwarzane są ww. produkty, czyli:

- 4 szlifierki,
- 2 wiertarki pionowe,
- 3 wiertarki poziome,
- 1 frezarka,
- 1 tokarka.

Dany produkt musi zostać poddany obróbce na ww. maszynach, aby mógł powstać. Czas potrzebny na daną obróbkę jest podany w treści zadania zgodnie z poniższą tabelą:

	P1	P2	P3	P4
Szlifowanie	0,4	0,6	-	-
Wiercenie pionowe	0,2	0,1	-	0,6
Wiercenie poziome	0,1	-	0,7	-
Frezowanie	0,06	0,04	-	0,05
Toczenie	-	0,05	0,02	-

Tabela.1 Czas produkcji 1 sztuki produktu podany w godzinach

Zgodnie z Tabelą.1 wiemy, że czas potrzebny na wytworzenie P1 to: 0,4+0,2+0,1+0,06 = 0,76 h, poprzez sumowanie jesteśmy w stanie obliczyć potrzebny czas na wytworzenie danego produktu.

Dochody ze sprzedaży produktów (w zł/sztukę) nie są podane wprost tylko trzeba je wyliczyć na podstawie podanych danych. Wektor R zawiera informację o cenie za dany produkt, gdzie wektor R opisuje 4-wymiarowy rozkład t-Studenta z 4 stopniami swobody, którego wartości zostały zawężone do przedziału [5; 12], a parametry μ i Σ charakteryzują się następująco:

$$\mu = \begin{pmatrix} 9 \\ 8 \\ 7 \\ 6 \end{pmatrix}, \Sigma = \begin{pmatrix} 16 & -2 & -1 & -3 \\ -2 & 9 & -4 & -1 \\ -1 & -4 & 4 & 1 \\ -3 & -1 & 1 & 1 \end{pmatrix}$$

Aby obliczyć dochód ze sprzedaży (w zł/sztukę) skorzystamy z dodatku dostarczonego przez prowadzącego „wo-tStudent.pdf", gdzie mamy opisane w jaki sposób obliczamy wartość oczekiwaną zawężonego rozkładu t-Studenta.

$$E(R) = \mu + \sigma \frac{\Gamma\left(\left(\frac{v-1}{2}\right)\left((v+a^2)^{-\left(\frac{v-1}{2}\right)} - (v+b^2)^{-\left(\frac{v-1}{2}\right)}\right)v^{\frac{v}{2}}\right.}{2\big(F_v(b) - F_v(a)\big)\Gamma(^{v}/_{2})\Gamma(^{1}/_{2})}$$

Z takiego wzoru możemy teraz obliczyć wartości dochodu ze sprzedaży P1, P2, P3 i P4. Wartości te obliczam automatycznie za pomocą kodu napisanego w języku Python. Poniżej w tabeli przedstawiam obliczone wartości z dokładnością do 4 miejsc po przecinku.

Produkt	Cena (w zł/sztukę)
P1	8.5796
P2	8.3440
P3	7.6828
P4	6.4757

Tabela. 2 Cena produktu za jedną sztukę

W zadaniu są wspomniane ograniczenia rynkowe na liczbę sprzedawanych produktów w danym miesiącu, jest to o tyle ważne, że potem podczas walidacji stworzonego modelu musimy uwzględnić te wartości, aby ocenić, czy zadanie zostało rozwiązane poprawnie.

	P1	P2	P3	P4
Styczeń	200	0	100	200
Luty	300	100	200	200
Marzec	0	300	100	200

Tabela.3 Ograniczenia rynkowe na sprzedaż danego produktu w danym miesiącu

Tabela. 3 jasno określa, ile maksymalnie możemy sprzedać danego produktu w danym miesiącu, nawet jeśli jest on najbardziej opłacalny pod względem dochodu oraz poświęconego czasu do jego wyprodukowania, nie możemy przekroczyć tych limitów.

Prowadzący wprowadził również ograniczenia, które musimy uwzględnić podczas tworzenia i walidacji modelu. Ograniczenia te dotycząc sprzedawania produktów, składowania ich w magazynie oraz czasu pracy w danym miesiącu, który jest stały.

Biorąc pod uwagę Tabelę.3 oraz ograniczenie, że jeśli sprzedajmy produkt P1 lub P2 to musimy sprzedawać produkt P4 w niemniejszej liczbie sztuk niż suma P1 i P2. Z tego wynika np. ograniczenia w Marcu, gdzie P1 nie może być sprzedawane, ale możemy wyprodukować 300 sztuk produktu P2 oraz 200 sztuk produktu P4, jeśli wyprodukujemy 100 sztuk produktu P2 to musimy wyprodukować co najmniej 100 sztuk produktu P4, który niekoniecznie nam się opłaca.

Druga część zadania jest podobna do pierwszego z kilka zmianami. Z modelu jednokryterialnego tworzymy model dwukryterialny, gdzie interesuje nas wartość zysku i ryzyka. Dane dotyczące scenariuszy generujemy z rozkładu t-student. Zmianie nie ulegają ograniczenia oraz stałe wartości dotyczące czasu produkcji, dostępności maszyn.
W przypadku mojego zadania zysk ma być mierzony za pomocą wartością średnią, a miara ryzyka za pomocą średniej różnicy Giniego. Zadanie drugie składa się z trzech podpunktów, w których mam wyznaczyć zbiór rozwiązań efektywnych, wskazać rozwiązania efektywne minimalnego ryzyka i maksymalnego zysku, a na końcu sprawdzić, czy zachodzi relacja dominacji stochastycznej pierwszego rzędu.

2. Określenie zmiennych decyzyjnych, ograniczeń i funkcji oceny.

2.1. Zadanie 1

W przypadku zmiennych decyzyjnych ograniczyłem się do trzech, które uważałem za najistotniejsze dla projektu. Interesowała mnie:
- produkcja danego produktu w danym miesiącu,
- sprzedaż danego produktu w danym miesiącu,
- przechowywanie danego produktu w danym miesiącu.

W ten sposób mogłem bezpośrednio sprawdzać czy produkcja, sprzedaż i przechowywanie produktów spełniają ograniczenia wynikające z treści zadania. Jednak bezpośrednio zysk będzie zależał od ilości sprzedanych produktów.

Posiadam cztery różne produktu: P1, P2, P3, P4.
Sprzedaż produktów odbywa się w ciągu trzech miesięcy: Styczeń, Luty i Marzec.

$production\_vars_{p,m}$ - zmienna całkowita reprezentująca produkcję danego produktu p w miesiącu m, gdzie $p \in \{P1, P2, P3, P4\}$, a $m \in \{Styczeń, Luty, Marzec\}$

$sales\_vars_{p,m}$ – zmienna całkowita reprezentująca sprzedaż danego produktu p w miesiącu m, gdzie $p \in \{P1, P2, P3, P4\}$, a $m \in \{Styczeń, Luty, Marzec\}$

$storage\_vars_{p,m}$ – zmienna całkowita reprezentująca przechowywanie danego produktu p w miesiącu m, gdzie $p \in \{P1, P2, P3, P4\}$, a $m \in \{Styczeń, Luty, Marzec\}$

Ograniczenia, które są wartościami stałymi to ilość dostępnych maszyn do produkcji produktów, czas potrzebny na wytworzenie danego produktu na danej maszynie, zostało to pokazane opisane na początku sprawozdania oraz w Tabeli.1.
Ograniczenia rynkowe na sprzedaż produktów:

$$sales\_vars_{p,m} \leq M_{p,m}, \forall_p \in \{P1, P2, P3, P4\}, \forall_m \in \{Styczeń, Luty, Marzec\}$$

, gdzie M to słownik z zawartymi danymi z Tabeli.3.

Ograniczenie dotyczące sprzedaży produktów P1, P2 oraz P4:

$$sales\_vars_{P4,m} \geq sales\_vars_{P1,m} + sales\_vars_{P2,m}, \forall_m \in \{Styczeń, Luty, Marzec\}$$

Ograniczenie dotyczące składowania do 200 sztuk każdego produktu:

$$storage\_vars_{p,m} \leq 200, \forall_p \in \{P1, P2, P3, P4\}, \forall_m \in \{Styczeń, Luty, Marzec\}$$

Ograniczenie dotyczące zachowania 50 sztuk każdego produktu pod koniec Marca:

$$storage\_vars_{p,Marzec} = 50, \forall_p \in \{P1, P2, P3, P4\}$$

Ograniczenie dotyczące dostępnego czasu pracy na maszynach w danym miesiącu:

$$\sum_{p \in \{P1,P2,P3,P4\}} a_{p,k} \cdot production\_vars_{p,m} \leq T_k, \forall_m \in \{Styczeń, Luty, Marzec\}$$

$$\forall_k \in \{Slifierka\ , Wiertarka\ Pionowa, Wiertarka\ Pozioma, Frezarka, Tokarka\}$$

, gdzie k to typ maszyny, a reprezentuje czas potrzebny na maszynie w celu stworzenia danego produktu, wartość T to całkowity dostępny czas na maszynach w danym miesiącu.

Funkcja celu, to maksymalizacja zysku ze sprzedaży produktów mając na uwadze koszty składowania produktów oraz inne ograniczenia w modelu:

$$\max\left(\sum_{m \in \{Styczeń,Luty,Marzec\}} \sum_{p \in \{P1,P2,P3,P4\}} (sales\_vars_{p,m} \cdot r_p - storage\_vars_{p,m} \cdot c_s) \right)$$

, gdzie r_p to zysk ze sprzedaży danego produktu (w zł/sztukę) wartości te są pokazane w Tabeli.2, c_s to koszt składowania produktu w magazynie (koszt ten jest stały i wynosi 1 zł/sztuka).

2.2. Zadanie 2

Zadanie 2 silnie bazuje na zadaniu pierwszym, ale zmienia się funkcja celu, dodajemy nowe ograniczenia i zmienne, które ułatwiają zamodelowanie problemu.

$profit\_vars_{p,m,scenarios}$ – zmienna ciągła reprezentująca zysk ze sprzedaży danego produktu p w miesiącu m dla danego scenariusza pomniejszona o koszty przechowywania produktów w magazynie, gdzie $p \in \{P1, P2, P3, P4\}$, $m \in \{Styczeń, Luty, Marzec\}$, a $scenarios \in \{1,\ 2, \dots, n\}$.

$d\_vars_{t_1, t_2, scenarios}$ – pomocnicza ciągła zmienna wprowadzona do modelu w celu obliczenia bezwzględnych różnic pomiędzy zyskami z różnych scenariuszy, gdzie $t_1 \in Scenarios,\ t_2 \in Scenarios, Scenarios \in \{1, 2, \dots, N\}$Jest ona potrzebna do implementacji średniej różnicy Giniego jako ryzyka w naszym modelu.
Ograniczenie dotyczące zmiennej d_vars zapewniające dodatnią wartość różnicy pomiędzy zyskami z poszczególnych scenariuszy.

$$d_{t_1,t_2} \geq profit\_vars_{t_1} - profit\_vars_{t_2}$$

$$d_{t_1,t_2} \geq profit\_vars_{t_2} - profit\_vars_{t_1}$$

, gdzie d_{t_1,t_2} jest naszą pomocniczą zmienną $d\_vars_{t_1, t_2}$, która jest zawsze dodatnia. Wartość $profit\_vars_{t_1}$ to zysk dla scenariusza t_1, a $profit\_vars_{t_2}$ to zysk dla scenariusza t_2.

Funkcja celu składa się z elementu zysku i elementu ryzyka.

Element zysku:

$$\sum_{t=1}^{n} p_t \cdot profit\_vars_t$$

, gdzie p_t to prawdopodobieństwo zajścia danego scenariusza, a $profit\_vars_t$ to zysk dla danego scenariusza.

Element ryzyka:

$$\lambda \cdot \sum_{t_1}^{n} \sum_{t_2}^{n} d\_vars_{t_1, t_2, scenarios}\, p_{t_1} p_{t_2}$$

, gdzie p_t to prawdopodobieństwo zajścia danego scenariusza, $d\_vars_{t_1, t_2, scenarios}$ to wartość bezwzględna różnicy zysku pomiędzy dwoma scenariuszami, a λ jest to współczynnik zamiany ryzyka na spodziewany zwrot i jest zwykle interpretowany jako współczynnik awersji do ryzyka.

Ostateczna funkcja celu:

$$max\left(\sum_{t=1}^{n} p_t \cdot profit\_vars_t - \lambda \cdot \sum_{t_1}^{n} \sum_{t_2}^{n} d\_vars_{t_1, t_2, scenarios}\, p_{t_1} p_{t_2}\right)$$

, gdzie p_t to prawdopodobieństwo zajścia danego scenariusza, $profit\_vars_t$ to zysk dla danego scenariusza, $d\_vars_{t_1, t_2, scenarios}$ to wartość bezwzględna różnicy zysku pomiędzy dwoma scenariuszami, a λ jest to współczynnik zamiany ryzyka na spodziewany zwrot i jest zwykle interpretowany jako współczynnik awersji do ryzyka.

3. Wskazanie i uzasadnienie przyjętych założeń.

3.1. Zadanie 1

Większość założeń wywodzi się bezpośrednio z treści zadania, nie ma większych luk logicznych, które trzeba byłoby wypełniać.

Rozpatruję przypadek, gdzie sprzedaż, produkcja i przechowywanie produktów odbywa się na przestrzeni trzech miesięcy, ze względu na liniowość problemu moglibyśmy stworzyć liniową zależność na resztę miesięcy w roku, ale nie jest to potrzebne do rozwiązania zadania.

Czasy produkcji są stałe i niezmienne w czasie, to samo tyczy się cen sprzedaży, ilości godzin w miesiącu, cen składowania towaru, limitów rynkowych. Ułatwia to zamodelowanie problemu.

Nie są to założenie idealnie odpowiadające warunkom prawdziwego życie, bo brakuje kosztów produkcji (materiały, prąd, i tym podobne elementy), pracowników, którzy są odpowiedzialni za produkcję, a tym samym kosztów zatrudnienia i prawdopodobnie wielu innych czynników, które wpłynęłyby ostatecznie na całość zadania, ale tworzę uproszczony model, który zawiera elementy wymienione w treści zadania.

3.2. Zadanie 2

Większość założeń do zadania drugiego pokrywa się z zadaniem pierwszym. Dodatkowym założeniem jest wygenerowanie N scenariuszy dla naszego problemu, które wynikają bezpośrednio z rozkładu t-Studenta. Scenariusze te są generowane losowo, ale dla powtarzalności testów ustawiamy jednego seeda, czyli dla każdego wywołania mojego kodu otrzymamy taki sam zestaw danych.

Scenariusze posiadają te same ograniczenia, te same parametry tworzenia produktów, te same dostępne maszyny, jedyna rzecz jaka ulega zmianie to cena za dany produkt w danym scenariuszu.

Wszystkie scenariusze mają jednakowe prawdopodobieństwo co będzie potrzebne przy kalkulacji zysku i ryzyka. W innym przypadku musiałbym generować prawdopodobieństwo dla każdego scenariusza. Zadanie na obecny moment jest wymagające obliczeniowo i nie ma potrzeby dokładać dodatkowych obliczeń w postaci różnych prawdopodobieństw.

Iterujemy po różnych wartościach λ, aby móc oszacować jak awersja do ryzyka wpływa na potencjalny zysk.

4. Wskazanie podstaw teoretycznych.

4.1. Zadanie 1

Do rozwiązania zadnia pierwszego korzystałem ze skryptów dostarczonych przez prowadzącego dotyczących rozkładu t-student oraz podstaw programowanie liniowego, podstaw modelowanie matematycznego z przedmiotu MOM, dokumentacji biblioteki PuLP. W zadaniu pierwszym głównie opierałem się na tym co było napisane w zadaniu, analizując treść, ograniczenia, a potem przekładałem to na kod. Po napisaniu kodu przeprowadziłem kilka testów, które potwierdzały jego poprawne działanie, czyli wartości zmiennych,

dostosowanie się do ograniczeń, czy każdy element kodu został wykorzystany i wzięty pod uwagę.

4.2. Zadanie 2

Do rozwiązania zadnia drugiego posłużyłem się skryptami profesora W.Ogarczyka dotyczącymi wspomagania decyzji w warunkach ryzyka, a dokładniej modelami M-R Markowitza. Profesor Ogarczyk opisuje tam problem optymalnego wyboru portfela inwestycyjnego korzystając z modelu Markowitza oraz zbiór dopuszczalny Q, w którym zmienne decyzyjne spełniają wyznaczone ograniczenia, co przekłada się na reprezentowanie portfela jako punktu w przestrzeni n-wymiarowej. Podstawą doboru optymalnego portfela jest maksymalizacja oczekiwanego zwrotu i minimalizacja ryzyka, która w moim przypadku jest definiowana przez średnią różnicę Giniego (GMD). Przykłady w dokumencie ilustrują różne podejścia do optymalizacji portfela, w tym użycie klasycznych zadań optymalizacji kwadratowej. Wyjaśniono również koncepcję efektywnego zestawu rozwiązań, gdzie każdy punkt reprezentuje portfel maksymalizujący oczekiwany zwrot przy danym poziomie ryzyka.

$$\Gamma = \frac{1}{2}\int\int |\xi - \eta| P_x(d\eta) = \frac{1}{2}\sum_{t'=1}^{T}\sum_{t''=1}^{T} |r_{t'} - r_{t''}| p_{t'} p_{t''}$$

Tak przedstawiona różnica Giniego nie nadaje się do zamodelowania w programowaniu liniowym ze względu na wprowadzenie nieliniowości przez wartość bezwzględną, ale profesor Ogarczyk wyprowadza w skrypcie różnicę Giniego w takiej postaci, żeby została ona użyta w problemach liniowych. Dzięki temu wyprowadzeniu mogłem odpowiednio zamodelować problem z zadania.

$$\max \sum_{t=1}^{T} p_t r_t - \lambda \sum_{t'=1}^{T} \sum_{t'' \neq t'} p_{t'} p_{t''} d_{t't''}$$

$$x \in Q_n$$

$$r_t = \sum_{j=1}^{n} r_{jt} x_j \; dla \; t = 1, \dots, T$$

$$d_{t't''} \geq r_{t'} - r_{t''}, d_{t't''} \geq r_{t'} \; dla \; t', t'' = 1, \dots, T; t'' \neq t'$$

W dokumencie opisano sposób weryfikacji, czy zachodzi relacja stochastyczna pierwszego rzędu (FSD - First Stochastic Dominance) między dwoma zmiennymi losowymi. Aby zmienna losowa X dominowała stochastycznie zmienną losową Y w sensie pierwszego rzędu, dystrybuanta zmiennej X musi być w każdym punkcie nie większa od dystrybuanty zmiennej Y, a przynajmniej w jednym punkcie musi być mniejsza.

Formalnie, X dominuje Y w sensie FSD, jeśli:

$$F_X(t) \leq F_Y(t) \; dla \; wszytskich \; t$$

, gdzie $F_X(t)\ i\ F_Y(t)$ to dystrybuanty odpowiednio dla X i Y. Dodatkowo, musi istnieć przynajmniej jeden punkt, w którym nierówność jest ostra.

5. Specyfikacja problemu decyzyjnego z dookreśleniem wszystkich elementów.

5.1. Zadanie 1

Oprócz podstawowych ograniczeń wynikających z zadania, należało dodać kilka oczywistych ograniczeń, które nie było zapisane w zadaniu.

Ograniczenie dotyczące zmiennych, żeby przyjmowały wartości nieujemne i były całkowite:

$$production\_vars_{p,m} \in Z^+, p \in \{P1, P2, P3, P4\}, m \in \{Styczeń, Luty, Marzec\}$$

$$sales\_vars_{p,m} \in Z^+, p \in \{P1, P2, P3, P4\}, m \in \{Styczeń, Luty, Marzec\}$$

$$storage\_vars_{p,m} \in Z^+, p \in \{P1, P2, P3, P4\}, m \in \{Styczeń, Luty, Marzec\}$$

Ograniczenie dotyczące sprzedaży, składowania i produkcji produktów, aby uwzględnić to w końcowej liczbie składowanych produktów w magazynie:

Ograniczenie dotyczące przechowywanie produktów w magazynie w styczniu:

$$storage\_vars_{p,Styczeń} = storage\_vars_0 + production\_vars_{p,Styczeń} - sales\_vars_{p,Styczeń}, \forall_p \in \{P1, P2, P3, P4\}$$

, gdzie $storage\_vars_{p,Styczeń}$ to ilość produktu p w magazynie z końcem stycznia, $storage\_vars_0$ to początkowa ilość produktu w magazynie, $production\_vars_{p,Styczeń}$ to produkcja produktu p w styczniu, $sales\_vars_p$ to sprzedaż produktu p w styczniu.

Ograniczenie dotyczące przechowywania produktów pod koniec każdego miesiąca, który nie jest styczniem:

$$storage\_vars_{p,m} = storage\_vars_{p,m-1} + production\_vars_{p,m} - sales\_vars_{p,m} \quad \forall_p \in \{P1, P2, P3, P4\}, \forall_m \in \{Luty, Marzec\}$$

Ograniczenie dotyczące sprzedaży produktów, aby nie przekroczyła ona ilości wyprodukowanych dóbr oraz tego co jest w magazynie w styczniu:

$$sales\_vars_{p,Styczeń} \leq production\_vars_{p,Styczeń} + storage\_vars_0, \forall_p \in \{P1, P2, P3, P4\}$$

, gdzie $storage\_vars_0$ to ilość produktu pod koniec grudnia.

Wszystkie te organiczna i funkcja celu powinny wystarczyć do poprawnego rozwiązania naszego zadania.

5.2. Zadanie 2

W celu poprawnego rozwiązania zadania drugiego musiałem zdefiniować kilka dodatkowych elementów.

$$profit\_vars_{p,m,scenarios} \in R^{+}, p \in \{P1, P2, P3, P4\}, m \in \{Styczeń, Luty, Marzec\}, Scenarios \in \{1, 2, \dots, N\}$$

$$d\_vars_{t_1, t_2, scenarios} \in R^{+}, p \in \{P1, P2, P3, P4\}, m \in \{Styczeń, Luty, Marzec\}, Scenarios \in \{1, 2, \dots, N\}$$

Reszta dodatkowych elementów jest ściśle związana z kodem, więc zaprezentuję i opiszę te elementy w punkcie 6.2 Zadanie 2.

6. Sformułowanie modelu w postaci do rozwiązania z wykorzystaniem wybranego narzędzia/środowiska implementacji (kompletny kod źródłowy).

Do napisania projektu użyłem języka programowania Python oraz biblioteki PuLP wraz z solverem CPLEX. Wybór ten jest podyktowany najlepszą znajomością języka, a solver został polecony przez prowadzącego.

6.1. Zadanie 1

W pierwszej kolejności należało obliczyć dochody ze sprzedaży produktów (w zł/sztukę) w tym celu skorzystałem z danych i metody podanej w punkcie 1. Analityczne sformułowanie modelu. Zainstalowałem i zaimportowałem odpowiednie biblioteki, które ułatwiły wykonywanie obliczeń. Podałem dane z zadania.

```
!pip install cplex
!pip install pulp

from scipy.special import gamma
from scipy.stats import t
import numpy as np
from math import sqrt

# Parameters of the Student's t-distribution
mu = np.array([9, 8, 7, 6])
Sigma = np.array([[16, -2, -1, -3],
                  [-2, 9, -4, -1],
                  [-1, -4, 4, 1],
                  [-3, -1, 1, 1]])

# Degrees of freedom
v = 4

# Alpha and Beta values
alpha = 5
```

```
beta = 12

# Empty array to store the results
E_values = []
```

Następnie automatycznie kalkulowałem wartości a, b oraz dochód ze sprzedaży produktów zaokrąglając wynik do czterech miejsc po przecinku.

```
# Iterate through mu and Sigma to calculate a, b, and E for each
component
for i in range(len(mu)):
    # Calculate a and b
    a = (alpha - mu[i]) / sqrt(Sigma[i,i])
    b = (beta - mu[i]) / sqrt(Sigma[i,i])

    # Calculate E
    E = mu[i] + sqrt(Sigma[i,i]) * (((gamma(v - 1) / 2) * ((v + a ** 2)
** (-(v - 1) / 2) - (v + b ** 2) ** (-(v - 1) / 2)) * (v ** (v / 2))) /
(2 * (t.cdf(b, v) - t.cdf(a, v)) * gamma(v / 2) * gamma(1 / 2)))

    # Append the result to the list of E values
    E_values.append(round(E, 4))

# Print the calculated values of E
print("Expected values of narrowed marginal distributions:")
print(E_values)
```

Tak obliczone wartości posłużą mi w dalszej części modelu.

Następnie zacząłem modelować problem z wykorzystaniem biblioteki PuLP, w tym celu dodałem wszystkie stałe wartości jak: produkty, miesiące, ilość dni w miesiącu, ilość zmian w ciągu dnia, ilość godzin w ciągu zmiany, ilość produktów z końcem grudnia, koszt przechowywania produktów w magazynie, pojemność magazynu, końcowa ilość produktu w marcu, czasy produkcji danego produktu, ograniczenia czasowe dla danej maszyny w postaci słownika, ograniczenia rynkowe w postaci słownika i dostępną ilość maszyn.

```
import pulp as pl

# Define the problem
model = pl.LpProblem("Maximize_Profit", pl.LpMaximize)

# Products and months
products = ['P1', 'P2', 'P3', 'P4']
months = ['Jan', 'Feb', 'Mar']

# Income from product sales in PLN/piece
product_income = {'P1': 8.5796, 'P2': 8.344, 'P3': 7.6828, 'P4':
6.4757}
```

```
# Time available per month (in hours)
days_per_month = 24
shifts_per_day = 2
hours_per_shift = 8
hours_per_month = days_per_month * shifts_per_day * hours_per_shift

# Initial stock and storage cost
initial_stock = 50
storage_cost_per_unit_per_month = 1
storage_capacity = 200
end_stock = 50

# Production times required per product in hours
production_times = {
    'P1': {'Grinding': 0.4, 'VerticalDrilling': 0.2,
'HorizontalDrilling': 0.1, 'Milling': 0.06, 'Lathing': 0},
    'P2': {'Grinding': 0.6, 'VerticalDrilling': 0.1,
'HorizontalDrilling': 0, 'Milling': 0.04, 'Lathing': 0.05},
    'P3': {'Grinding': 0, 'VerticalDrilling': 0, 'HorizontalDrilling':
0.7, 'Milling': 0, 'Lathing': 0.02},
    'P4': {'Grinding': 0, 'VerticalDrilling': 0.6,
'HorizontalDrilling': 0, 'Milling': 0.05, 'Lathing': 0}
}

# Market limits for sales of Products
market_limits = {
    "P1": {"Jan": 200, "Feb": 300, "Mar": 0},
    "P2": {"Jan": 0, "Feb": 100, "Mar": 300},
    "P3": {"Jan": 100, "Feb": 200, "Mar": 100},
    "P4": {"Jan": 200, "Feb": 200, "Mar": 200},
}

# Machines available
machines = {"Grinding": 4, "VerticalDrilling": 2, "HorizontalDrilling":
3, "Milling": 1, "Lathing": 1}
```

W kolejnym kroku zdefiniowałem trzy zmienne, które odzwierciedlają realia sprzedaży, produkcji i przechowywanie produktów. Biblioteka PuLP pozwala wprowadzić ograniczenia bezpośrednio przy deklaracji zmiennej bez konieczności deklarowania ich osobno.

```
# Variables for production, sales, and storage of products each month
production_vars = pl.LpVariable.dicts("Production", ((product, month)
for product in products for month in months), lowBound=0,
cat='Integer')
sales_vars = pl.LpVariable.dicts("Sales", ((product, month) for product
in products for month in months), lowBound=0, cat='Integer')
```

```
storage_vars = pl.LpVariable.dicts("Storage", ((product, month) for
product in products for month in months), lowBound=0,
upBound=storage_capacity, cat='Integer')
```

Przedostatnim krokiem było zawarcie ograniczeń dla naszego problemu. Ograniczenia te pokrywają się z ograniczeniami z punktu 2.1.

```
# Constrain for P4 sales based on P1 and P2 sales
for month in months:
  model += sales_vars['P4', month] >= sales_vars['P1', month] +
sales_vars['P2', month]

# Inventory and sales constraints
for product in products:
    model += storage_vars[product, 'Jan'] == initial_stock +
production_vars[product, 'Jan'] - sales_vars[product, 'Jan']  # for
January
    for i in range(1, len(months)):
        month = months[i]
        prev_month = months[i-1]
        model += storage_vars[product, month] == storage_vars[product,
prev_month] + production_vars[product, month] - sales_vars[product,
month]  # for rest of the months
    model += storage_vars[product, 'Mar'] == end_stock  # Ending stock

# Machine time constraints for each product and month
for month in months:
    total_production_time = sum(production_vars[product, month] *
production_times[product][machine]
                                for product in products
                                for machine in machines)
    model += total_production_time <= hours_per_month,
f"Global_production_time_{month}"

# Market sales constraints
for product in products:
    for month in months:
        model += sales_vars[product, month] <=
market_limits[product][month], f"Max_sales_{product}_{month}"
        model += sales_vars[product, month] <= production_vars[product,
month] + (storage_vars[product, months[months.index(month) - 1]] if
month != 'Jan' else initial_stock)
```

Ostatni element całego modelu to funkcja celu oraz wypisanie wszystkich wartości zmiennych dla danych miesięcy. Funkcja celu to nic innego jak dochód ze sprzedaży danego produktu pomniejszony o koszty przechowywanie produktów.

```
# Objective function: Maximize profit from sales minus storage costs
model += pl.lpSum([sales_vars[product, month] * product_income[product]
- storage_vars[product, month] * storage_cost_per_unit_per_month
```

```
                    for product in products for month in months]),
"Total_Profit"
```

Rozwiązujemy problem i wyświetlamy wyniki.

```
# Solve the model
model.solve()

# Check the status of the solution and print the results
print("Status:", pl.LpStatus[model.status])
if pl.LpStatus[model.status] == 'Optimal':
    print("Total Profit:", pl.value(model.objective))

    # Printing headers
    print(f"{'Product':<10} {'Month':<10} {'Production':<12}
{'Sales':<10} {'Storage':<10}")

    # Display the results in tabular form
    for product in products:
        for month in months:
            production = production_vars[product, month].varValue
            sales = sales_vars[product, month].varValue
            storage = storage_vars[product, month].varValue
            print(f"{product:<10} {month:<10} {production:<12}
{sales:<10} {storage:<10}")

    print("\nProduction Time Usage per Product and Month:")
    for month in months:
        print(f"\nMonth: {month}")
        for product in products:
            total_production_time = sum(
                production_vars[product, month].varValue *
production_times[product].get(machine, 0)
                for machine in machines.keys()
            )
            print(f"  Product: {product}, Total Production Time:
{total_production_time} hours")

else:
    print("No optimal solution was found.")
```

6.2. Zadanie 2

Pierwsza zmiana względem zadania 1 to implementacja generacji scenariuszy, która wygląda następująco.

```
import pulp as pl
import numpy as np
from scipy.stats import multivariate_t
import matplotlib.pyplot as plt

path_to_cplex =
r'/Applications/CPLEX_Studio2211/cplex/bin/arm64_osx/cplex'
solver = pl.CPLEX_CMD(path=path_to_cplex)

# Parameters
mu = np.array([9, 8, 7, 6])
Sigma = np.array([[16, -2, -1, -3],
                  [-2, 9, -4, -1],
                  [-1, -4, 4, 1],
                  [-3, -1, 1, 1]])
v = 4  # Degrees of freedom

# Alpha and Beta values for clipping
alpha = 5
beta = 12

# Number of scenarios
n_scenarios = 20

# Seed for reproducibility
np.random.seed(0)

# Random samples from multivariate t-distribution
t_dist = multivariate_t(loc=mu, shape=Sigma, df=v)
samples = t_dist.rvs(size=n_scenarios)

# Clip the values
scenarios = np.clip(samples, alpha, beta)
```

Wartości macierzy, stopnie swobody, alpha, beta nie ulegają zmianie. Zmienna n_sceanrios jest odpowiedzielana za ilość wygenerowanych scenariuszy, np.random.seed(0) to wartość, która pozwala generować dokładnie takie same wartości dla danego scenariusza niezależnie od przeprowadzanych testów. Następnie wykorzystanie biblioteki scipy.stats i funkcji multivariate_t genereuje nasz rozkład, z którego wybieramy scenariusze, które są odpowiednio przycinane ze względu na wartości alpha i beta.

Kod jest podobny do kodu z pierwszego zadania, prócz dodanych elementów takich jak zmienne reprezentujące zysk w danym scenariuszu oraz pomocnicza zmienna do kalkulacji wartości bezwględnej różnicy zysku pomiędzy scenariuszami.

```
# Profit variables for each scenario
profit_vars = [pl.lpSum(sales_vars[product, month] *
scenarios[t][list(products).index(product)] for product in products for
month in months) - pl.lpSum(storage_vars[product, month] *
storage_cost_per_unit_per_month for product in products for month in
months) for t in range(n_scenarios)]

# Variables for Gini differences
d_vars = pl.LpVariable.dicts("D", ((t1, t2) for t1 in
range(n_scenarios) for t2 in range(t1+1, n_scenarios)), lowBound=0,
cat='Continuous')
```

Stworzenie zmiennej kalkulującej prawdopodbieństwo zajścia danego scenariusza. Nasz przypadek jest uproszczony ze względu na równe prawdopodobieństwa.

```
# Probability of each scenario
probabilities = np.ones(n_scenarios) / n_scenarios
```

Następnie zadeklarowałem wartość zmiennej lambda, po której będziemy iterować i sprawdzać jak zysk i ryzyko mają się dla naszego modelu oraz listę na przechowywanie rezlutatów.

```
# Loop over lambda_values
lambda_values = np.linspace(0, 50, num=50)
results = []
```

Następnie w pętli odbywa się kalulacja zysku i ryzyka.

```
for lambda_value in lambda_values:
    objective_profit = pl.lpSum(probabilities[t] * profit_vars[t] for t
in range(n_scenarios))
    objective_risk =  pl.lpSum(probabilities[t1] * probabilities[t2] *
d_vars[min(t1, t2), max(t1, t2)] for t1 in range(n_scenarios) for t2 in
range(n_scenarios) if t1 != t2)

    # Objective
    current_objective = objective_profit - lambda_value *
objective_risk

    # Set the current objective in the model
    model.setObjective(current_objective)

    # Solve the model
    model.solve(solver)
```

```
    # Store results
    total_profit = pl.value(objective_profit)
    total_risk = pl.value(objective_risk)
    results.append((total_profit, total_risk, lambda_value))
```

Wyniki są dołączane do naszej listy.

Następnie za pomocą biblioteki matplolib wykorzystując zebrane dane jestem wstanie narysować zebrane wartości na wykresie.

```
# Plotting the results
profits, risks, lambdas = zip(*results)
plt.figure(figsize=(10, 6))
plt.scatter(risks, profits, c=lambdas, cmap='viridis')
plt.colorbar(label='Lambda Value')
plt.title('Risk-Reward Frontier')
plt.xlabel('Total Risk')
plt.ylabel('Total Profit')
plt.grid(True)
plt.show()
```

Następnie w celu znalezienia rozwiązań efektywnych minimlanego ryzka i maksymlanego zysku napisałem trzy oddzielne funkcje, które wykorzystują siebie nawzajem. Funkcja is_dominated porównuje ze sobą wyniki naszego modelu, szukając takich kandyadtów, którzy mają zysk większy bądź równy oraz ryzyko mniejsze bądź równe do innego kandydata. Zwracamy True albo False w zależności od spełeniania tych warunków.

```
def is_dominated(candidate, others):
    for other in others:
        if (other[0] >= candidate[0] and other[1] <= candidate[1]) and
(other[0] > candidate[0] or other[1] < candidate[1]):
            return True
    return False
```

W funkcji find_extreme_pareto_solutions szukam zbioru opymlanych rozwiązań ze wszystkich wygenrowanych scenariuszy. W pierwszej kolejności sortuję wyniki od tych, które dały najwyższy zysk, incializuję pustą listę dla zbioru optymlanych rozwiązań. Wykorzystując poprzednią funkcję generuję szukany zbiór. W kolejnym kroku znajduję rozwiązanie efektywne minimlanego ryzyka i maksymlanego zysku.

```
def find_extreme_pareto_solutions(results):
    pareto_optimal = []
    min_risk = float('inf')
    max_profit = float('-inf')
    min_risk_solution = None
    max_profit_solution = None

    for candidate in sorted(results, key=lambda x: x[0], reverse=True):
        if not is_dominated(candidate, pareto_optimal):
```

```
            pareto_optimal.append(candidate)
            if candidate[1] < min_risk:
                min_risk = candidate[1]
                min_risk_solution = candidate
            if candidate[0] > max_profit:
                max_profit = candidate[0]
                max_profit_solution = candidate

    return pareto_optimal, min_risk_solution, max_profit_solution
```

Na sam koniec wywołuję wszystkie funkcje i wypisuję odpowiednie wartości.

```
# Calculate Pareto optimal solutions and the extremes
pareto_solutions, min_risk_sol, max_profit_sol =
find_extreme_pareto_solutions(results)

print("Pareto Optimal Solutions:")
for solution in pareto_solutions:
    print(f"Profit: {solution[0]}, Risk: {solution[1]}, Lambda:
{solution[2]}")

print("\nMinimum Risk Solution:")
print(f"Profit: {min_risk_sol[0]}, Risk: {min_risk_sol[1]}, Lambda:
{min_risk_sol[2]}")

print("\nMaximum Profit Solution:")
print(f"Profit: {max_profit_sol[0]}, Risk: {max_profit_sol[1]}, Lambda:
{max_profit_sol[2]}")
```

7. Omówienie testów poprawności implementacji oraz omówienie wyników.
7.1. Zadanie 1

W celu sprawdzenia poprawności implementacji modelu zdecydowałem się na manipulację wartościami dotyczącymi ceny produktów, ograniczeń rynkowych, godzin pracy i pojemności magazynu. Uwzględniając te zmiany i ograniczenia w modelu powinienem móc stwierdzić, czy implementacja jest poprawna i nie zachodzą, żadne dziwne anomalie.

Wynik bez żadnych zmian:
Podstawą do oceny działania modelu jest znalezienie optymalnego rozwiązania używając danych z zadania bez żadnych zmian, dopiero na jego podstawie będziemy mogli oszacować czy nowe wyniki po zmienieniu pewnych danych są zgodne z logiką i naszymi oczekiwaniami.

```
Status: Optimal
Total Profit: 11494.179999999998
Product    Month      Production   Sales      Storage
P1         Jan        209.0        200.0      59.0
P1         Feb        141.0        200.0      0.0
P1         Mar        50.0         0.0        50.0
P2         Jan        101.0        0.0        151.0
P2         Feb        0.0          0.0        151.0
P2         Mar        99.0         200.0      50.0
P3         Jan        50.0         100.0      0.0
P3         Feb        202.0        200.0      2.0
P3         Mar        148.0        100.0      50.0
P4         Jan        150.0        200.0      0.0
P4         Feb        202.0        200.0      2.0
P4         Mar        248.0        200.0      50.0

Production Time Usage per Product and Month:

Month: Jan
  Product: P1, Total Production Time: 158.84 hours
  Product: P2, Total Production Time: 79.78999999999999 hours
  Product: P3, Total Production Time: 36.0 hours
  Product: P4, Total Production Time: 97.5 hours

Month: Feb
  Product: P1, Total Production Time: 107.16000000000001 hours
  Product: P2, Total Production Time: 0.0 hours
  Product: P3, Total Production Time: 145.43999999999997 hours
  Product: P4, Total Production Time: 131.29999999999998 hours

Month: Mar
  Product: P1, Total Production Time: 38.0 hours
  Product: P2, Total Production Time: 78.21 hours
  Product: P3, Total Production Time: 106.55999999999999 hours
  Product: P4, Total Production Time: 161.2 hours
```

Jak widać dla każdego produktu (P1, P2, P3, P4) otrzymujemy wartości dotyczące produkcji, sprzedaży i przechowywania w magazynie (pod koniec miesiąca) dla danego miesiąca.
Jak widać w styczniu produkujemy P1w ilości 209 sztuk i sprzedajemy 200 sztuk, ponieważ zostało nam 50 sztuk w magazynie z grudnia, sprzedaliśmy prawie wszystko co wyprodukowaliśmy, więc zostaje nam w magazynie 59 sztuk. Podobna sytuacja występuje z

P3 w styczniu oraz dla P4, gdzie sprzedaliśmy wszystko łącznie z zapasami z magazynu. Wszystkie ograniczenie rynkowe pokrywają się z aktualnymi sprzedażami, więc model pod tym względem działa poprawnie. Zysk ze sprzedaży również wygląda prawidłowo.

Zmiana cen produktów:
Zamiana cen produktów jest problematyczna ze względu na powiązania pomiędzy produktami występującymi w treści zadania. Produkty P1, P2 i P4 są ze sobą powiązane ograniczeniami w sprzedaży, więc niezależnie od ich ceny ograniczenie to musi zostać spełnione.

- Zmiana ceny produktu P1 na 0:

```
Status: Optimal
Total Profit: 8990.739999999998
Product    Month      Production   Sales      Storage
P1         Jan        0.0          50.0       0.0
P1         Feb        0.0          0.0        0.0
P1         Mar        50.0         0.0        50.0
P2         Jan        66.0         0.0        116.0
P2         Feb        139.0        100.0      155.0
P2         Mar        95.0         200.0      50.0
P3         Jan        50.0         100.0      0.0
P3         Feb        200.0        200.0      0.0
P3         Mar        150.0        100.0      50.0
P4         Jan        150.0        200.0      0.0
P4         Feb        200.0        200.0      0.0
P4         Mar        250.0        200.0      50.0

Production Time Usage per Product and Month:

Month: Jan
  Product: P1, Total Production Time: 0.0 hours
  Product: P2, Total Production Time: 52.14 hours
  Product: P3, Total Production Time: 36.0 hours
  Product: P4, Total Production Time: 97.5 hours

Month: Feb
  Product: P1, Total Production Time: 0.0 hours
  Product: P2, Total Production Time: 109.81 hours
  Product: P3, Total Production Time: 144.0 hours
  Product: P4, Total Production Time: 130.0 hours

Month: Mar
  Product: P1, Total Production Time: 38.0 hours
  Product: P2, Total Production Time: 75.05 hours
  Product: P3, Total Production Time: 108.0 hours
  Product: P4, Total Production Time: 162.5 hours
```

Jak widzimy wartości produkcji, sprzedaży i przechowywanie produktu P1 uległy zmianie we wszystkich miesiącach. Jest to pożądane zachowanie ze względu na brak dochodów z tego produktu, a wymienione wcześniej ograniczenie jest spełnione za pomocą produktu P2. Spada zysk co jest oczekiwane w przypadku obniżenia zysku z produktu.

- **Zmiana ceny produktu P3 na 0:**

Tutaj zachowanie będzie podobne do tego jak w przypadku wyżej z P1.

- **Zmiana ceny produktu P4 na 0:**

W tym przypadku zaobserwujemy najciekawszą rzecz, ponieważ sprzedaż produktu P4 jest silnie uzależniona od sprzedaży P1 i P2, więc nawet jeśli cena sprzedaży będzie równa zero, to i tak będziemy sprzedawać ten produkt w dużych ilościach.

```
Status: Optimal
Total Profit: 7608.759999999999
Product    Month      Production   Sales      Storage
P1         Jan        209.0        200.0      59.0
P1         Feb        141.0        200.0      0.0
P1         Mar        50.0         0.0        50.0
P2         Jan        101.0        0.0        151.0
P2         Feb        0.0          0.0        151.0
P2         Mar        99.0         200.0      50.0
P3         Jan        50.0         100.0      0.0
P3         Feb        202.0        200.0      2.0
P3         Mar        148.0        100.0      50.0
P4         Jan        150.0        200.0      0.0
P4         Feb        202.0        200.0      2.0
P4         Mar        248.0        200.0      50.0

Production Time Usage per Product and Month:

Month: Jan
  Product: P1, Total Production Time: 158.84 hours
  Product: P2, Total Production Time: 79.78999999999999 hours
  Product: P3, Total Production Time: 36.0 hours
  Product: P4, Total Production Time: 97.5 hours

Month: Feb
  Product: P1, Total Production Time: 107.16000000000001 hours
  Product: P2, Total Production Time: 0.0 hours
  Product: P3, Total Production Time: 145.43999999999997 hours
  Product: P4, Total Production Time: 131.29999999999998 hours

Month: Mar
  Product: P1, Total Production Time: 38.0 hours
  Product: P2, Total Production Time: 78.21 hours
  Product: P3, Total Production Time: 106.55999999999999 hours
  Product: P4, Total Production Time: 161.2 hours
```

Jak widać produkcja P1, P2 i P4 jest na wysokim poziomie, wszystko przez organicznie, że sprzedaż P4 nie może być mniejsza niż suma sprzedaży P1 i P2. Zdecydowanie odbija się to na dochodzie.

Zmiana ograniczeń rynkowych:

Ograniczenia rynkowe w istotny sposób wpływają na końcowy zysk ze sprzedaży produktów, w pierwszym teście sprawdziłem, że wszystkie ograniczenia rynkowe są odpowiednio spełnione. Zwiększę limity sprzedaży dla najdroższego produktu oczekując jego zwiększonej produkcji i sprzedaży, ale jeśli będzie to dotyczyć produktu P1, P2 lub P4 to niekoniecznie wartość ta będzie dużo wyższa ze względu na niewielkie różnice w cenie, wymagany czas na przygotowanie oraz ograniczenia.

- **Zmiana dla P1 we wszystkich miesiącach do 500**

```
Status: Optimal
Total Profit: 11549.299999999997
Product    Month      Production   Sales      Storage
P1         Jan        306.0        200.0      156.0
P1         Feb        143.0        200.0      99.0
P1         Mar        151.0        200.0      50.0
P2         Jan        0.0          0.0        50.0
P2         Feb        0.0          0.0        50.0
P2         Mar        0.0          0.0        50.0
P3         Jan        50.0         100.0      0.0
P3         Feb        200.0        200.0      0.0
P3         Mar        150.0        100.0      50.0
P4         Jan        150.0        200.0      0.0
P4         Feb        202.0        200.0      2.0
P4         Mar        248.0        200.0      50.0

Production Time Usage per Product and Month:

Month: Jan
  Product: P1, Total Production Time: 232.56 hours
  Product: P2, Total Production Time: 0.0 hours
  Product: P3, Total Production Time: 36.0 hours
  Product: P4, Total Production Time: 97.5 hours

Month: Feb
  Product: P1, Total Production Time: 108.68 hours
  Product: P2, Total Production Time: 0.0 hours
  Product: P3, Total Production Time: 144.0 hours
  Product: P4, Total Production Time: 131.29999999999998 hours

Month: Mar
  Product: P1, Total Production Time: 114.76000000000002 hours
  Product: P2, Total Production Time: 0.0 hours
  Product: P3, Total Production Time: 108.0 hours
  Product: P4, Total Production Time: 161.2 hours
```

- **Zmiana dla P3 we wszystkich miesiącach do 500**

```
Status: Optimal
Total Profit: 12015.9026
Product    Month      Production   Sales      Storage
P1         Jan        136.0        186.0      0.0
P1         Feb        151.0        151.0      0.0
P1         Mar        50.0         0.0        50.0
P2         Jan        0.0          0.0        50.0
```

```
P2          Feb         0.0           45.0        5.0
P2          Mar         45.0          0.0         50.0
P3          Jan         267.0         317.0       0.0
P3          Feb         197.0         197.0       0.0
P3          Mar         386.0         336.0       50.0
P4          Jan         136.0         186.0       0.0
P4          Feb         196.0         196.0       0.0
P4          Mar         50.0          0.0         50.0

Production Time Usage per Product and Month:

Month: Jan
  Product: P1, Total Production Time: 103.36000000000001 hours
  Product: P2, Total Production Time: 0.0 hours
  Product: P3, Total Production Time: 192.23999999999998 hours
  Product: P4, Total Production Time: 88.39999999999999 hours

Month: Feb
  Product: P1, Total Production Time: 114.76000000000002 hours
  Product: P2, Total Production Time: 0.0 hours
  Product: P3, Total Production Time: 141.83999999999997 hours
  Product: P4, Total Production Time: 127.39999999999999 hours

Month: Mar
  Product: P1, Total Production Time: 38.0 hours
  Product: P2, Total Production Time: 35.55 hours
  Product: P3, Total Production Time: 277.92 hours
  Product: P4, Total Production Time: 32.5 hours
```

Zgodnie z założeniami produkcja i sprzedaż produktu P3 jest faworyzowana w tym momencie ze względu na możliwości sprzedażowe, które są prawie dwukrotnie wyższy. Produkt P3 wymaga zdecydowanie mniej czasu na wytworzenie, a nawet jest on najniższy ze wszystkich produktów. Model zachowuje się poprawnie.

Zmiana ilości godzin pracy:
Wydłużenie czasu pracy (godzin w jednej zmianie) powinno mieć wpływ na wyprodukowanie większej ilości produktów, a zmniejszenie ilości godzin powinno oznaczać zmniejszenie ilości produktów i spadek dochodu. Ale musimy znieść ograniczenia rynkowe w celu czerpania korzyści z większej ilości godzin.

- **Zmiana godzin w jednej zmianie z 8 na 16 godzin, limit sprzedaży 500 sztuk dla każdego produktu w każdym miesiącu**

```
Status: Optimal
Total Profit: 24370.7839
Product     Month       Production    Sales       Storage
P1          Jan         396.0         446.0       0.0
P1          Feb         488.0         488.0       0.0
P1          Mar         485.0         435.0       50.0
P2          Jan         0.0           50.0        0.0
P2          Feb         0.0           0.0         0.0
P2          Mar         50.0          0.0         50.0
P3          Jan         246.0         296.0       0.0
P3          Feb         111.0         111.0       0.0
P3          Mar         62.0          12.0        50.0
P4          Jan         446.0         496.0       0.0
```

```
P4          Feb         488.0         488.0       0.0
P4          Mar         485.0         435.0       50.0

Production Time Usage per Product and Month:

Month: Jan
  Product: P1, Total Production Time: 300.96000000000004 hours
  Product: P2, Total Production Time: 0.0 hours
  Product: P3, Total Production Time: 177.11999999999998 hours
  Product: P4, Total Production Time: 289.9 hours

Month: Feb
  Product: P1, Total Production Time: 370.88 hours
  Product: P2, Total Production Time: 0.0 hours
  Product: P3, Total Production Time: 79.91999999999999 hours
  Product: P4, Total Production Time: 317.2 hours

Month: Mar
  Product: P1, Total Production Time: 368.6 hours
  Product: P2, Total Production Time: 39.5 hours
  Product: P3, Total Production Time: 44.64 hours
  Product: P4, Total Production Time: 315.25 hours
```

Zgodnie z przewidywanymi wydłużony czas pracy wpłynął na zwiększenie produkcji i dochodów.

- **Zmiana godzin w jednej zmianie z 8 na 4 godziny, limit sprzedaży 500 sztuk dla każdego produktu w każdym miesiącu**

```
Status: Optimal
Total Profit: 5921.697499999998
Product     Month       Production    Sales       Storage
P1          Jan         108.0         158.0       0.0
P1          Feb         128.0         128.0       0.0
P1          Mar         77.0          27.0        50.0
P2          Jan         0.0           50.0        0.0
P2          Feb         0.0           0.0         0.0
P2          Mar         50.0          0.0         50.0
P3          Jan         10.0          60.0        0.0
P3          Feb         16.0          16.0        0.0
P3          Mar         61.0          11.0        50.0
P4          Jan         158.0         208.0       0.0
P4          Feb         128.0         128.0       0.0
P4          Mar         77.0          27.0        50.0

Production Time Usage per Product and Month:

Month: Jan
  Product: P1, Total Production Time: 82.08000000000001 hours
  Product: P2, Total Production Time: 0.0 hours
  Product: P3, Total Production Time: 7.2 hours
  Product: P4, Total Production Time: 102.7 hours

Month: Feb
  Product: P1, Total Production Time: 97.28 hours
  Product: P2, Total Production Time: 0.0 hours
  Product: P3, Total Production Time: 11.52 hours
```

```
  Product: P4, Total Production Time: 83.2 hours

Month: Mar
  Product: P1, Total Production Time: 58.52 hours
  Product: P2, Total Production Time: 39.5 hours
  Product: P3, Total Production Time: 43.919999999999995 hours
  Product: P4, Total Production Time: 50.05 hours
```

Zgodnie z przewidywanymi skrócony czas pracy wpłynął na znaczące zmniejszenie produkcji i dochodów.

Zmiana pojemności magazynu:

W przypadku zmian pojemności magazynu nie przewiduję większych zmian, przechowywanie większej ilości produktów w magazynie to koszt, którego nie chcemy ponosić, a ograniczenia rynkowe nie pozwalają sprzedać nam większej ilości produktu.

- **Zmiana pojemności magazynu z 200 sztuk na 1000**

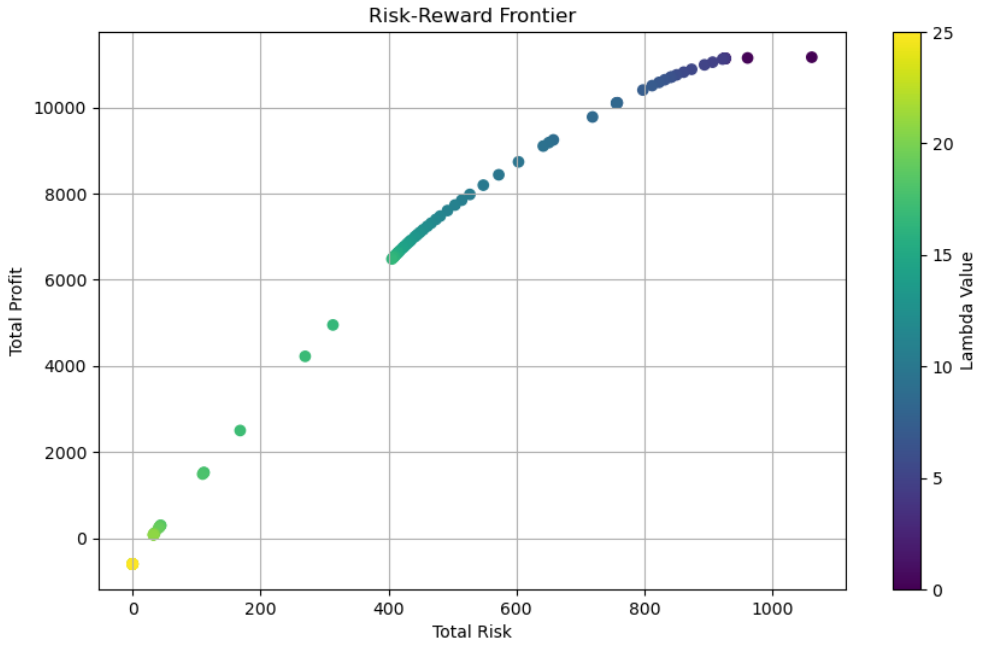
Tak jak zakładałem zmiana pojemności magazynu nie wpływa na nasz zysk, wynik jest identyczny do podstawowego wyniku.

7.3. Zadanie 2

Najważniejsze dla mnie w zadaniu drugim było sprawdzenie jak zachowuje się nasz model w zależności od awersji do ryzyka.

Zadanie drugie jest trochę bardziej wymagające obliczeniowo od zadania pierwszego, dlatego ograniczyłem zakres generowanych scenariuszy do maksymalnie 100 oraz generowanie wartości lambda do 50 z różnym krokiem, ale generowanie takiej ilości danych jest bardzo czasochłonne, więc do testów będę używał mniejszych wartości.

Wynik bez żadnych zmian 100 scenariuszy, lambda od 0 do 25 z 100 punktami:



Jak możemy zauważyć znalezione rozwiązania dla wygenerowanych scenariuszy wyglądają jak na powyższym wykresie. Na pionowej osi „Total Profit" mamy całkowity zysk dla danego ryzyka, które zależy od wartości lambda, który informuje nas o awersji do ryzyka. Im awersja do ryzyka większa, tym mniej będziemy go podejmować. Jak widać im większe ryzyko tym większy zysk, a im mniejsze ryzyko tym mniejszy zysk, co jest logicznym rozumowaniem.

Dla minimalnego ryzyka, które wynosi zero nie podejmujemy żadnego ryzyka, a to oznacza, że nie chcemy ryzykować produkcją i sprzedażą, jeśli z góry nie wiemy, że na tym zarobimy. Z tego powodu nie produkujemy i nie sprzedajemy żadnych produktów, jedyne co robimy to przechowujemy początkowe 50 sztuk każdego produktu od grudnia do marca w celu realizacji ograniczenia w postaci 50 sztuk każdego produktu na koniec miesiąca. W ten sposób jedyne co generujemy to koszt przechowywania produktów, który wynosi 1 zł za każdą sztukę produktu. Mamy 4 produkty, każdego w ilości pięćdziesięciu sztuk przez trzy miesiące, więc końcowy koszt to: 600 zł. I dokładnie taką wartość otrzymuję rozwiązując model dla ryzyka o wartości zero, całkowity zysk to -600.

Dla maksymalnego ryzyka sytuacja jest odwrotna, ponieważ podejmując większe ryzyko możemy oczekiwać większych zysków, ale również większych start. Wartość dla

maksymalnego ryzyka powinna być zbliżona do wyniku z pierwszego zadania, ale nie będzie identyczna ze względu na wygenerowanie wielu scenariuszy z wieloma różnymi cenami za dany produkt. W tym wypadku osiągnęliśmy wartość zysku na poziomie: 11166.

Minimum Risk Solution:

Profit: -600.0000000000005, Risk: 0.0, Lambda: 20.959595959595962

Maximum Profit Solution:

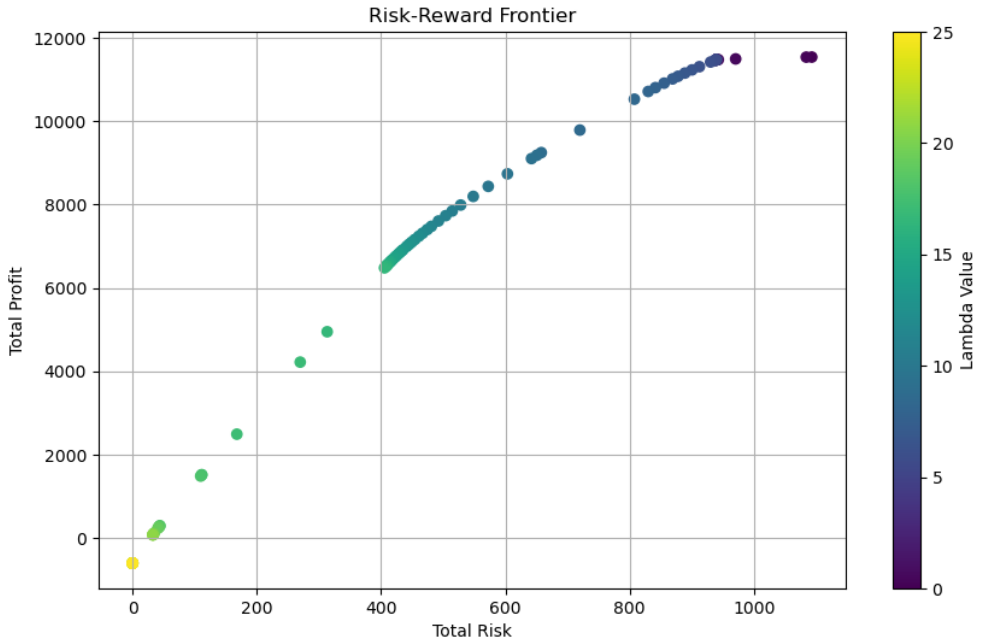
Profit: 11166.022633457791, Risk: 1061.5165956258284, Lambda: 0.0

W tym zadaniu zmiana cen nie jest możliwa, ale możemy zmienić inne parametry naszego zadania i zobaczyć, czy model zachowuje się zgodnie z przewidywaniami.

Zmiana ograniczeń rynkowych:

Tak jak w poprzednim przypadku zmiana ograniczeń rynkowych wpłynie na wynik końcowy, ale tylko dla największego ryzyka. Wartość dla najmniejszego ryzyka przy reszcie ograniczeń nie powinna ulec zmianie, chyba że dodalibyśmy nowe ograniczenia wymuszające produkcję, przechowywanie większej ilości produktów w magazynie lub zniesienie kosztów przechowywyana.

- **Zmiana dla P3 we wszystkich miesiącach do 500, 100 scenariuszy, lambda od 0 do 25 z 50 punktami:**



Minimum Risk Solution:

Profit: -600.0000000000005, Risk: 0.0, Lambda: 20.959595959595962

Maximum Profit Solution:

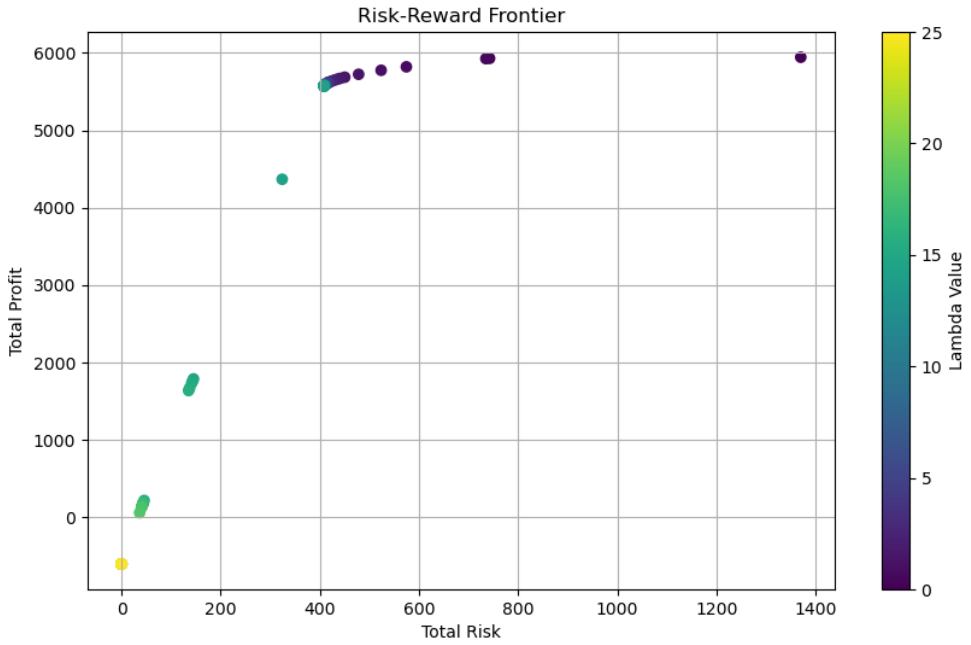
Profit: 11537.663816678025, Risk: 1092.86335481677, Lambda: 0.0

Zgodnie z założeniami model zachowuje się poprawnie, faworyzując produkcję produktu P3, który ma największe limity sprzedaży. Przy większych ilościach sprzedaży rośnie zysk, ale również ryzyko.

Zmiana ilości godzin pracy:
Wydłużenie czasu pracy (godzin w jednej zmianie) powinno mieć wpływ na wyprodukowanie większej ilości produktów, a zmniejszenie ilości godzin powinno oznaczać zmniejszenie ilości produktów i spadek dochodu. Ale musimy znieść ograniczenia rynkowe w celu czerpania korzyści z większej ilości godzin.

- **Zmiana godzin w jednej zmianie z 8 na 4 godziny, limit sprzedaży 300 sztuk dla każdego produktu w każdym miesiącu, 100 scenariuszy, lambda od 0 do 25 z 50 punktami:**

Niestety ze względu na wydajność komputera oraz małej optymalizacji kodu, nie jestem w stanie ustawić limitu sprzedaży na 500 sztuk, ponieważ CPLEX rozwiązuje model bardzo długo co uniemożliwia przeprowadzenia kilku testów w sensownym czasie. Limit sprzedaży ustawiłem na 300.



Minimum Risk Solution:

Profit: -600.0000000000001, Risk: 0.0, Lambda: 18.434343434343436

Maximum Profit Solution:

Profit: 5945.320386712527, Risk: 1369.8316048724257, Lambda: 0.0

Model zachowuje się poprawnie, mimo wyższych limitów sprzedaży, ograniczony czas na produkcję nie pozwala na wykorzystanie ich w całości. Stąd niższy zysk, a ryzyko zbliżone do poprzedniego testu.