Rotman

Master of Management Analytics

DATATHON 2022

Team Violet

June 15, 2022 Prepared by Sameer Pasha, Sky Niu, Sunny Chen, David Hamilton

Rotman School of Management
UNIVERSITY OF TORONTO

Rotman

Objective

Selecting five power play specialists and four penalty kill specialists to help Toronto Six assemble a team for their new NWHL season.

Power Play Specialists:

- The main objective of the power play players are to score goals and pass precisely.
- We are aiming to build a team that is not only good at the above skills but also display a versatile performance and implement a strong strategy.

Penalty Kill Specialists:

- The main objective of penalty kill players is to prevent opponent from scoring goals.
- The key skills of penalty killer are takeaways, precise passing and puck recovery.

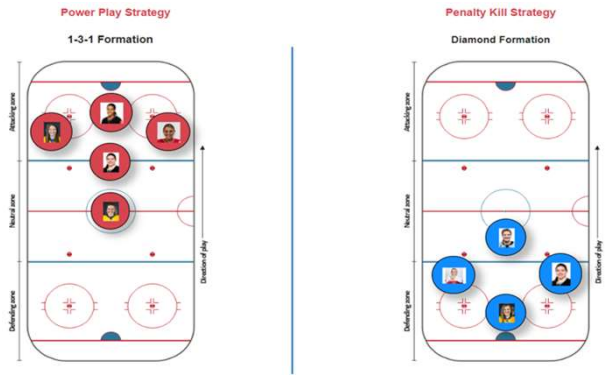
Strategies

Power Play:

- The strategy that is selected for power play is a 1-3-1 player formation.
- The 1-3-1 can spread out players inside the zone, keep a player in front of the net at all times, and position shooters in crucial positions to shoot.

Penalty kill:

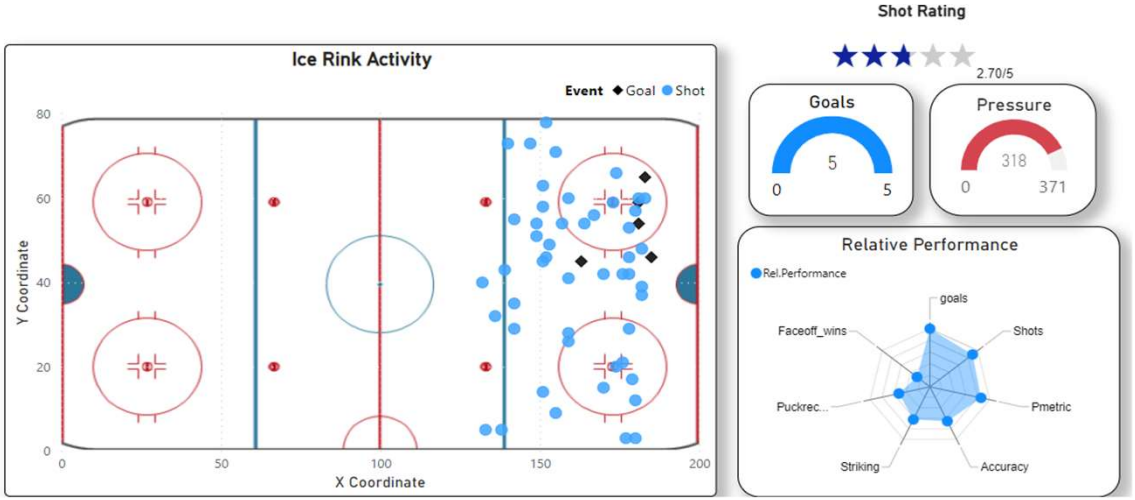
- Diamond penalty kill formation is the one going to be implemented in this experiment.
- The player at the apex is a takeaway expert and the right and left wing are players who are defenders and passers.
- The player at the back is a puck recovery specialist who is also good at the above skills, therefore the team will gain possession quickly after an unsuccessful shot attempt is taken.



Note: Please feel free to access below our Power BI dashboard to interact with player performances and strategies.

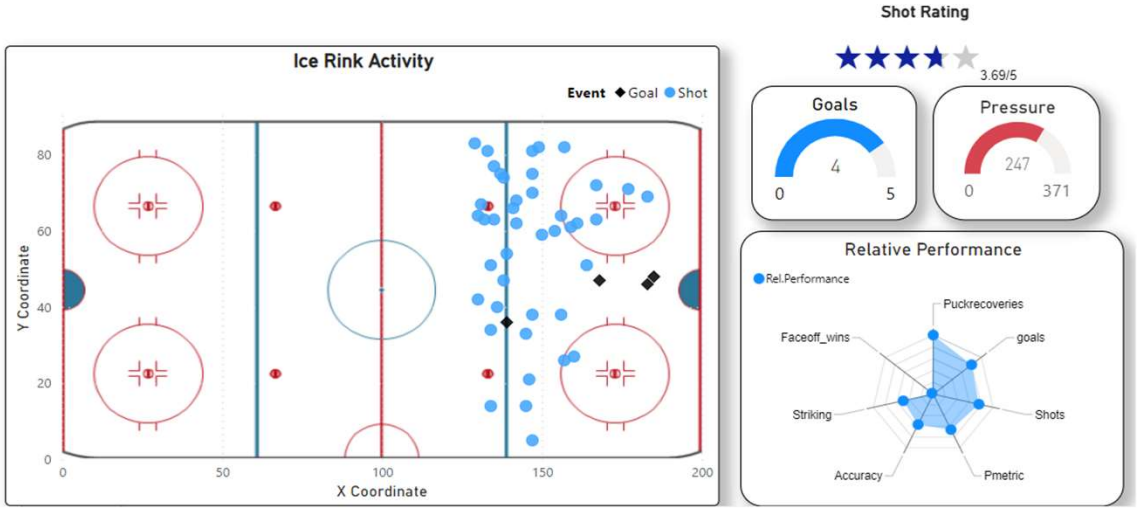
Rotman

Power Play Specialists



Mikyla Grant-Mentis

- She is the **highest** goal scorer with 5 goals in the last season.
- She has the **highest** number of shots on net.
- She is an offensive player, who can act as a center forward in the 1-3-1 formation.
- Her shots were majorly in the center of the field and she has the third highest pressure metric.
- Her pass accuracy with Taylor woods is **87.5%**.

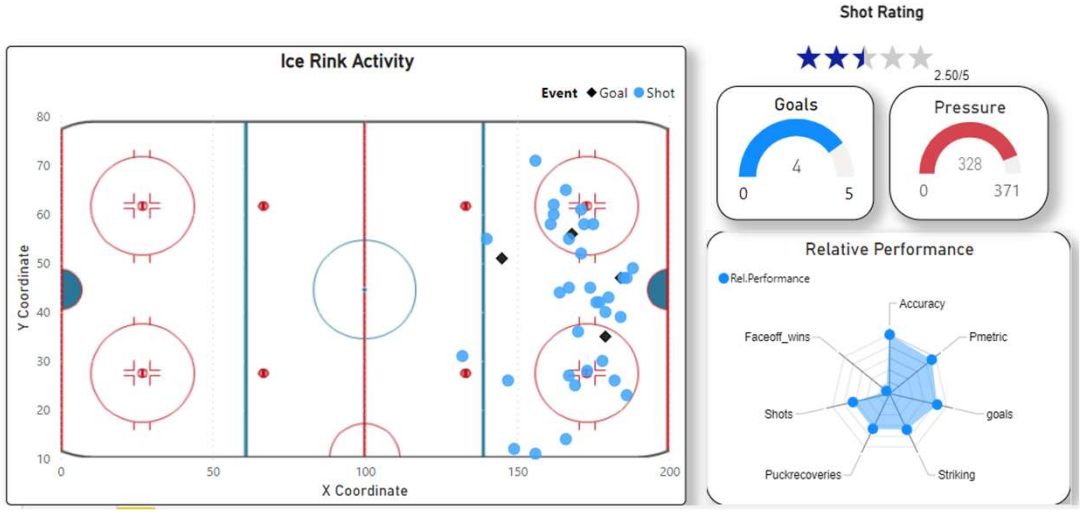


Taylor Woods

- She recovered the second highest number of loose pucks.
- She is a shooting expert who scored 4 goals in the last season.
- She has the third highest number of shots on net.
- Most of her shots were around the blue line, and she has the sixth highest pressure metric.
- She is a good shooter with advanced puck recovery skills, who can be a bumper in the 1-3-1 formation.

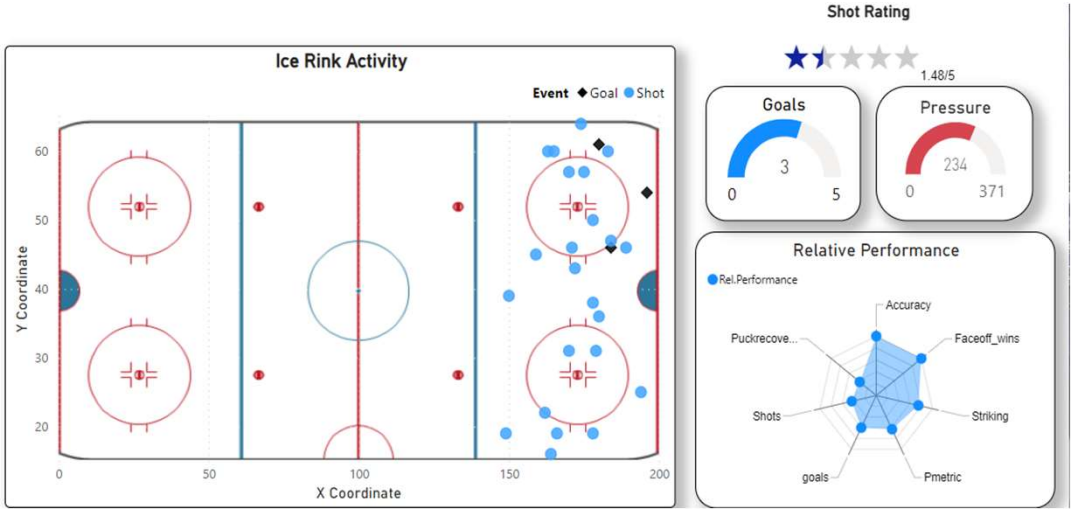
Rotman

Power Play Specialists



Samantha Davis

- She is a good shooter who scored 4 goals in the last season.
- Her shots were majorly in the center of the field, and she has the second highest pressure metric.
- Her pass accuracy in the last season is 72.5%.
- She has expertise in precise passing and striking pucks, who can act as a right flank in the 1-3-1 formation.

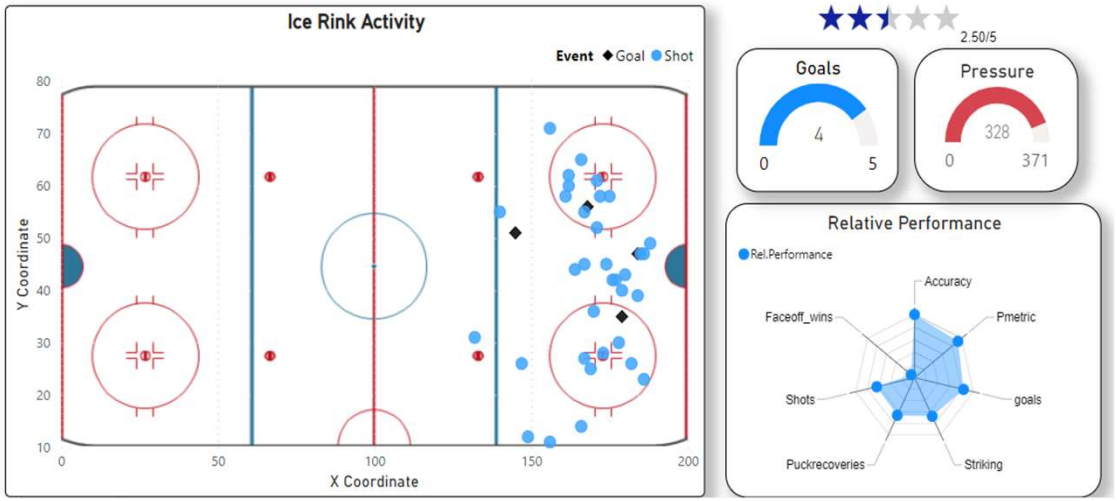


Jillian Dempsey

- She has the **highest** number of total faceoff wins in the last season.
- Her pass accuracy in the last season is 71.6%.
- Her shots spread out in the field, and she has the eighth highest pressure metric.
- She has expertise in precise passing and striking pucks, who can act as a left flank in the 1-3-1 formation.

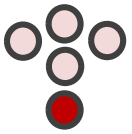
Rotman

Power Play Specialists



Mckenna Brand

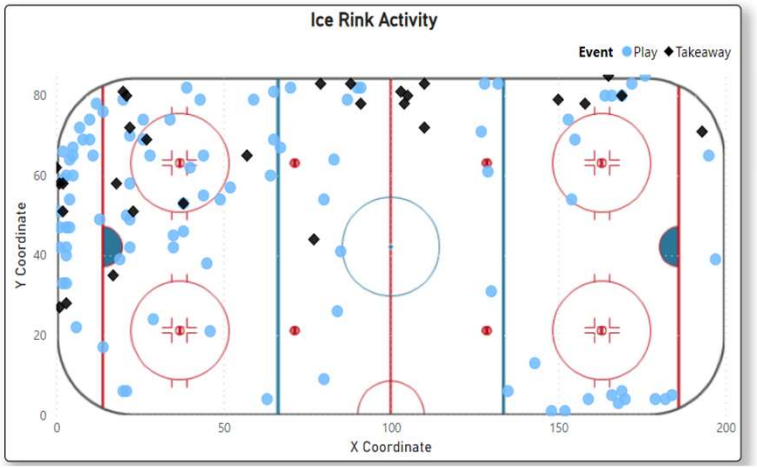
- Her shots were majorly in the center of the field, and she has the **highest** pressure metric.
- She is a good shooter who scored 3 goals in the last season.
- Her pass accuracy in the last season is 69.9%.
- She has expertise in precise passing and striking pucks, who can act as a quarterback in the 1-3-1 formation.

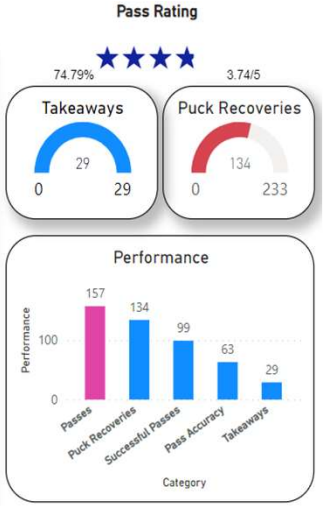


- Shot rating of a player is determined by the nature of the shots that the player has played. If a player has more ‘one timer’ and ‘traffic shots’ then the player has more shot rating.
- A Radar chart is used to visualize the relative performance of a player. Relative performance of a player in a particular category is determined by
  the players performance compared to the highest performance in that particular category taken as a percentile.

Rotman

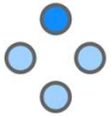
Penalty Kill Specialists

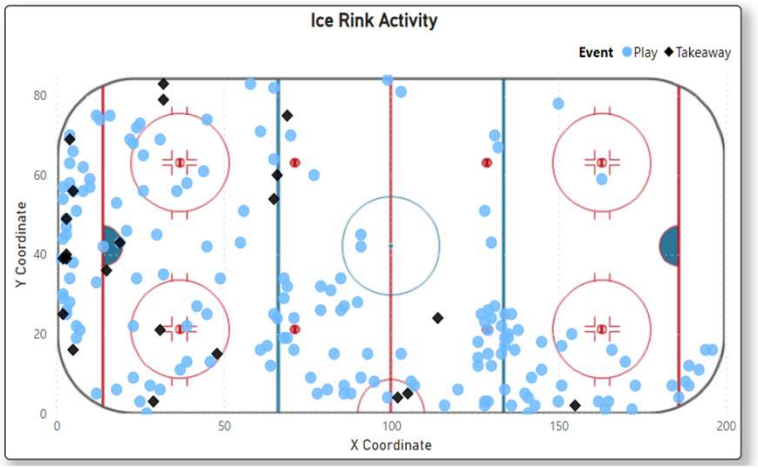


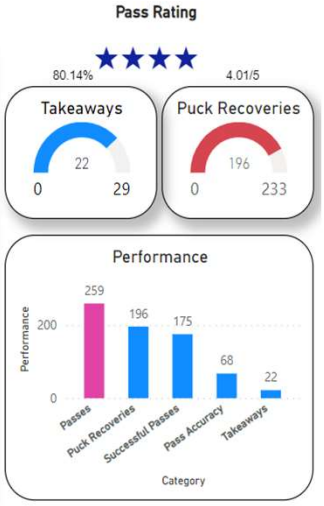


Dominique Kremer

- Dominque is a **takeaway specialist** and the league leader in total takeaways.
- She also excels at puck recovery, furthering her team's control of the puck to run out the clock.
- Her pass rating in the dataset is 74.79%.
- She plays at the top of the diamond to maximize her abilities.





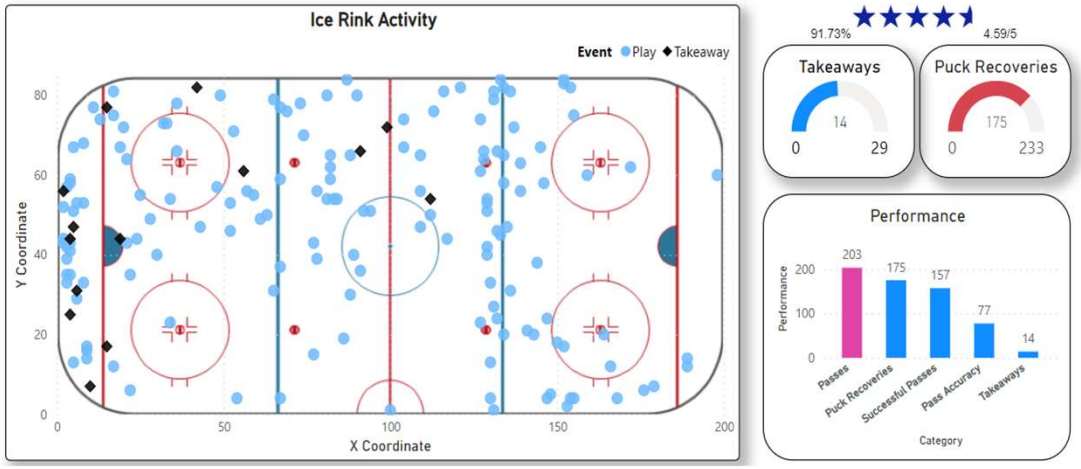


Mallory Souliotis

- Mallory is a **puck recovery specialist** who consistently beats the other team to the puck.
- Her pass rating in the dataset is 80.14%.
- She also excels at takeaways, with a focus the defensive zone and behind the net.
- She plays at the bottom of the diamond as the last line of defence before the goalie.

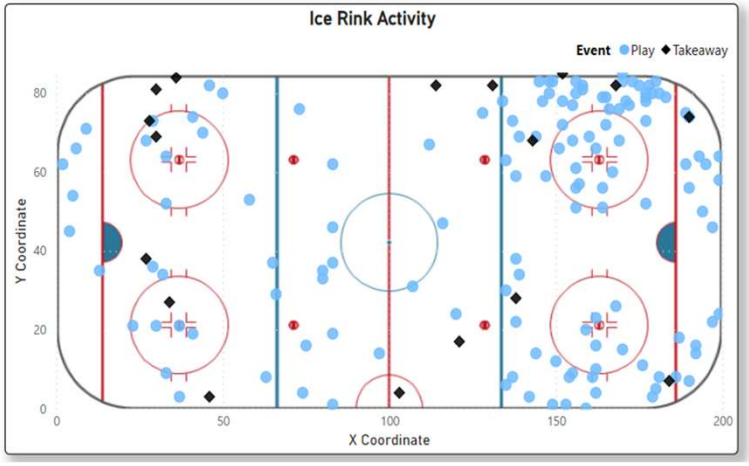
Rotman

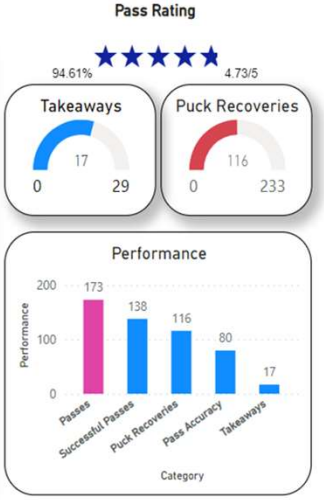
Penalty Kill Specialists



Taylor Woods

- Taylor is a **passing specialist** who racks up big passing numbers.
- Her pass rating of 91.73% is 4th highest in the dataset, with more passes than any player ahead of her.
- She also excels at puck recovery, ranking 3rd in the dataset.
- She plays on the right side of the diamond, using her precision passing to keep control of the puck.





Tereza Vanisova

- Tereza is the other **passing specialist** with the second highest pass rating in the league.
- Her pass rating in the dataset is 94.61%.
- She also has good numbers for takeaways and puck recoveries to help her team control the puck.
- She plays on the left side of the diamond and uses her skills to run out the clock until the penalty ends.

Rotman

Appendix

Linear Programming

- The Selection of the team is an optimization problem subject to constraints. It can be treated as a Linear Programming problem having Decision variables, Objective Function and Constraints. Excel Solver is used to implement and solve this problem.

Decision Variables:

- Each of the selected players among the pool are considered as decision variables.
- The decision variables are of binary data type. They take values 0 (not selected) or 1 (selected).The notation of decision variables is x1,x2,x3…….

Power Play:

- Objective Function
  - To maximize goals (g1*x1+g2*x2…….g20*x20). The objective function is designed in such a way that the top scorers get more weightage.
  - Players who have at least **two** goals are selected.
- Constraints
  - **Five** players are needed in the team among the pool(X1+X2……..+X20 = 5)
  - At least **three** players must be **excellent goal scorers** (X1+X2…….+X20 >=3)
  - **One** player from top 2 **puck recovery specialists** (X4+X7 = 1)
  - **One** player from top 2 **pressure specialists** (X3+X8=1)

Appendix

Penalty Kill:

- Objective Function:
  - To maximize Takeaways (t1*x1+t2*x2……..t54*x54). Top 50 %ile of players sorted by takeaways are taken for selection.
  - The objective function is designed in such a way that players with more takeaways get more advantage.
- Constraints:
  - **Four** players are needed in the team among the pool (X1+X2…….+X54 =4)
  - At least **two takeaway** specialists among top 25% (X1+X2…….+X30 >= 2)
  - **Two** excellent **passers** among top 4 passers (X23+X37+X38+X45 = 2)
  - **One puck recovery** specialist among top 2 puck recovery specialists (X8+X24 =1)

- The solving method used for solving this problem is the Simplex Linear Programming as all the constraints and the objective function are linear in nature.
- Tools used in the project
  - R (Data Analysis).
  - Excel Solver (Linear Programming).
  - Power BI (Data visualization for dashboard).

Note: Please feel free to access below our Power BI dashboard to interact with player performances and strategies.

Rotman

Appendix

- Penalty kill R code:

```
library(dplyr)


##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
NWHL <- read.csv("Rotman MMA Summer Datathon NWHL.csv", header = TRUE)
```

```
Takeaway_January23 <- filter(NWHL,Event == "Takeaway",game_date == "2021-01-23")

Takeaway_January24 <- filter(NWHL,Event == "Takeaway",game_date == "2021-01-24")

Takeaway_January26 <- filter(NWHL,Event == "Takeaway",game_date == "2021-01-26")

Takeaway_January27 <- filter(NWHL,Event == "Takeaway",game_date == "2021-01-27")

Takeaway_January30 <- filter(NWHL,Event == "Takeaway",game_date == "2021-01-30")

Takeaway_January31 <- filter(NWHL,Event == "Takeaway",game_date == "2021-01-31")

Takeaway_February01 <- filter(NWHL,Event == "Takeaway",game_date == "2021-02-01")
```

```
January23 = Takeaway_January23 %>% group_by(Player)  %>%
                    summarise(total_takeaway_Jan23 = length(Event))

January24 = Takeaway_January24 %>% group_by(Player)  %>%
                    summarise(total_takeaway_Jan24 = length(Event))

January26 = Takeaway_January26 %>% group_by(Player)  %>%
                    summarise(total_takeaway_Jan26 = length(Event))

January27 = Takeaway_January27 %>% group_by(Player)  %>%
                    summarise(total_takeaway_Jan27 = length(Event))

January30 = Takeaway_January30 %>% group_by(Player)  %>%
                    summarise(total_takeaway_Jan30 = length(Event))

January31 = Takeaway_January31 %>% group_by(Player)  %>%
                    summarise(total_takeaway_Jan31 = length(Event))

February01 = Takeaway_February01 %>% group_by(Player)  %>%
                    summarise(total_takeaway_Feb01 = length(Event))
```

```
Player_merge1 <- merge(January23,January24,by="Player",all = TRUE)
Player_merge2 <- merge(Player_merge1,January26,by="Player",all = TRUE)
Player_merge3 <- merge(Player_merge2,January27,by="Player",all=TRUE)
Player_merge4 <- merge(Player_merge3,January30,by="Player",all=TRUE)
Player_merge5 <- merge(Player_merge4,January31,by="Player",all=TRUE)
Player_merge_final <- merge(Player_merge5,February01,by="Player",all=TRUE)
```

```
takeaway_sum = rowSums(Player_merge_final[2:8],na.rm = TRUE)
Total_game <- 7
Game_played <- Total_game - rowSums(is.na(Player_merge_final))
Player_takeaway_list <- cbind(Player_merge_final,takeaway_sum,Game_played)
```

```
Player_takeaway_list$takeaway_per_game <- takeaway_sum/Game_played
Final_result_takeaway <- Player_takeaway_list[order(Player_takeaway_list$takeaway_per_game,
                                                    decreasing = TRUE),]
```

```
Play_January23 <- filter(NWHL,Event == "Play" | Event == "Incomplete Play",
                         game_date == "2021-01-23")

Play_January24 <- filter(NWHL,Event == "Play" | Event == "Incomplete Play",
                         game_date == "2021-01-24")

Play_January26 <- filter(NWHL,Event == "Play" | Event == "Incomplete Play",
                         game_date == "2021-01-26")

Play_January27 <- filter(NWHL,Event == "Play" | Event == "Incomplete Play",
                         game_date == "2021-01-27")

Play_January30 <- filter(NWHL,Event == "Play" | Event == "Incomplete Play",
                         game_date == "2021-01-30")

Play_January31 <- filter(NWHL,Event == "Play" | Event == "Incomplete Play",
                         game_date == "2021-01-31")

Play_February01 <- filter(NWHL,Event == "Play" | Event == "Incomplete Play",
                          game_date == "2021-02-01")


successful_Play_January23 <- filter(NWHL,Event == "Play",
                                    game_date == "2021-01-23")
```


Rotman

```
successful_Play_January24 <- filter(NWHL,Event == "Play",
                                    game_date == "2021-01-24")

successful_Play_January26 <- filter(NWHL,Event == "Play",
                                    game_date == "2021-01-26")

successful_Play_January27 <- filter(NWHL,Event == "Play",
                                    game_date == "2021-01-27")

successful_Play_January30 <- filter(NWHL,Event == "Play",
                                    game_date == "2021-01-30")

successful_Play_January31 <- filter(NWHL,Event == "Play",
                                    game_date == "2021-01-31")

successful_Play_February01 <- filter(NWHL,Event == "Play",
                                     game_date == "2021-02-01")
```

```
January23_pass = Play_January23 %>% group_by(Player)  %>%
                    summarise(total_pass_Jan23 = length(Event))

January24_pass = Play_January24 %>% group_by(Player)  %>%
                    summarise(total_pass_Jan24 = length(Event))

January26_pass = Play_January26 %>% group_by(Player)  %>%
                    summarise(total_pass_Jan26 = length(Event))

January27_pass = Play_January27 %>% group_by(Player)  %>%
                    summarise(total_pass_Jan27 = length(Event))

January30_pass = Play_January30 %>% group_by(Player)  %>%
                    summarise(total_pass_Jan30 = length(Event))

January31_pass = Play_January31 %>% group_by(Player)  %>%
                    summarise(total_pass_Jan31 = length(Event))

February01_pass = Play_February01 %>% group_by(Player)  %>%
                    summarise(total_pass_Feb01 = length(Event))


successful_January23_pass = successful_Play_January23 %>% group_by(Player)  %>%
                    summarise(total_pass_Jan23_successful = length(Event))

successful_January24_pass = successful_Play_January24 %>% group_by(Player)  %>%
                    summarise(total_pass_Jan24_successful = length(Event))

successful_January26_pass = successful_Play_January26 %>% group_by(Player)  %>%
                    summarise(total_pass_Jan26_successful = length(Event))

successful_January27_pass = successful_Play_January27 %>% group_by(Player)  %>%
                    summarise(total_pass_Jan27_successful = length(Event))

successful_January30_pass = successful_Play_January30 %>% group_by(Player)  %>%
                    summarise(total_pass_Jan30_successful = length(Event))

successful_January31_pass = successful_Play_January31 %>% group_by(Player)  %>%
                    summarise(total_pass_Jan31_successful = length(Event))

successful_February01_pass = successful_Play_February01 %>% group_by(Player)  %>%
                    summarise(total_pass_Feb01_successful = length(Event))
```

```
Player_merge1_play <- merge(January23_pass,January24_pass,by="Player",all = TRUE)
Player_merge2_play <- merge(Player_merge1_play,January26_pass,by="Player",all = TRUE)
Player_merge3_play <- merge(Player_merge2_play,January27_pass,by="Player",all=TRUE)
Player_merge4_play <- merge(Player_merge3_play,January30_pass,by="Player",all=TRUE)
Player_merge5_play <- merge(Player_merge4_play,January31_pass,by="Player",all=TRUE)
Player_merge6_play <- merge(Player_merge5_play,February01_pass,by="Player",all=TRUE)

Player_merge1_play_successful <- merge(successful_January23_pass,
                                       successful_January24_pass,
                                       by="Player",all = TRUE)
Player_merge2_play_successful <- merge(Player_merge1_play_successful,
                                       successful_January26_pass,
                                       by="Player",all = TRUE)
Player_merge3_play_successful <- merge(Player_merge2_play_successful,
                                       successful_January27_pass,
                                       by="Player",all=TRUE)
Player_merge4_play_successful <- merge(Player_merge3_play_successful,
                                       successful_January30_pass,
                                       by="Player",all=TRUE)
Player_merge5_play_successful <- merge(Player_merge4_play_successful,
                                       successful_January31_pass,
                                       by="Player",all=TRUE)
Player_merge6_play_successful <- merge(Player_merge5_play_successful,
                                       successful_February01_pass,
                                       by="Player",all=TRUE)


Player_pass_merge_final <- merge(Player_merge6_play,Player_merge6_play_successful,
                                 by = "Player",all=TRUE)
```

```
pass_sum = rowSums(Player_pass_merge_final[2:8],na.rm = TRUE)
successful_pass_sum = rowSums(Player_pass_merge_final[9:15],na.rm = TRUE)
Player_pass_list <- cbind(Player_pass_merge_final,pass_sum,successful_pass_sum)
```

```
Player_pass_list$pass_successful_rate <- successful_pass_sum/pass_sum
Final_result_pass <- Player_pass_list[order(Player_pass_list$pass_successful_rate,
                                            decreasing = TRUE),]
```

```
puckrecovery_January23 <- filter(NWHL,Event == "Puck Recovery",game_date == "2021-01-23")

puckrecovery_January24 <- filter(NWHL,Event == "Puck Recovery",game_date == "2021-01-24")

puckrecovery_January26 <- filter(NWHL,Event == "Puck Recovery",game_date == "2021-01-26")
```

```
puckrecovery_January27 <- filter(NWHL,Event == "Puck Recovery",game_date == "2021-01-27")

puckrecovery_January30 <- filter(NWHL,Event == "Puck Recovery",game_date == "2021-01-30")

puckrecovery_January31 <- filter(NWHL,Event == "Puck Recovery",game_date == "2021-01-31")

puckrecovery_February01 <- filter(NWHL,Event == "Puck Recovery",game_date == "2021-02-01")
```

```
January23_puckrecovery = puckrecovery_January23 %>% group_by(Player)  %>%
                    summarise(total_pukerecovery_January23 = length(Event))

January24_puckrecovery = puckrecovery_January24 %>% group_by(Player)  %>%
                    summarise(total_pukerecovery_January23 = length(Event))

January26_puckrecovery = puckrecovery_January26 %>% group_by(Player)  %>%
                    summarise(total_pukerecovery_January23 = length(Event))

January27_puckrecovery = puckrecovery_January27 %>% group_by(Player)  %>%
                    summarise(total_pukerecovery_January23 = length(Event))

January30_puckrecovery = puckrecovery_January30 %>% group_by(Player)  %>%
                    summarise(total_pukerecovery_January23 = length(Event))

January31_puckrecovery = puckrecovery_January31 %>% group_by(Player)  %>%
                    summarise(total_pukerecovery_January23 = length(Event))

February01_puckrecovery = puckrecovery_February01 %>% group_by(Player)  %>%
                    summarise(total_pukerecovery_January23 = length(Event))
```

```
Player_merge1_puckrecovery <- merge(January23_puckrecovery,
                                    January24_puckrecovery,
                                    by="Player",all = TRUE)
Player_merge2_puckrecovery <- merge(Player_merge1_puckrecovery,January26_puckrecovery,
                                    by="Player",all = TRUE)
Player_merge3_puckrecovery <- merge(Player_merge2_puckrecovery,January27_puckrecovery,
                                    by="Player",all=TRUE)
```

```
## Warning in merge.data.frame(Player_merge2_puckrecovery,
## January27_puckrecovery, : column names 'total_pukerecovery_January23.x',
## 'total_pukerecovery_January23.y' are duplicated in the result
```

```
Player_merge4_puckrecovery <- merge(Player_merge3_puckrecovery,January30_puckrecovery,
                                    by="Player",all=TRUE)
```

```
## Warning in merge.data.frame(Player_merge3_puckrecovery,
## January30_puckrecovery, : column names 'total_pukerecovery_January23.x',
## 'total_pukerecovery_January23.y' are duplicated in the result
```

```
Player_merge5_puckrecovery <- merge(Player_merge4_puckrecovery,January31_puckrecovery,
                                    by="Player",all=TRUE)
```

```
## Warning in merge.data.frame(Player_merge4_puckrecovery,
## January31_puckrecovery, : column names 'total_pukerecovery_January23.x',
## 'total_pukerecovery_January23.y', 'total_pukerecovery_January23.x',
## 'total_pukerecovery_January23.y' are duplicated in the result
```

```
Player_merge_final_puckrecovery <- merge(Player_merge5_puckrecovery,February01_puckrecovery,
                                         by="Player",all=TRUE)
```

```
## Warning in merge.data.frame(Player_merge5_puckrecovery,
## February01_puckrecovery, : column names 'total_pukerecovery_January23.x',
## 'total_pukerecovery_January23.y', 'total_pukerecovery_January23.x',
## 'total_pukerecovery_January23.y' are duplicated in the result
```

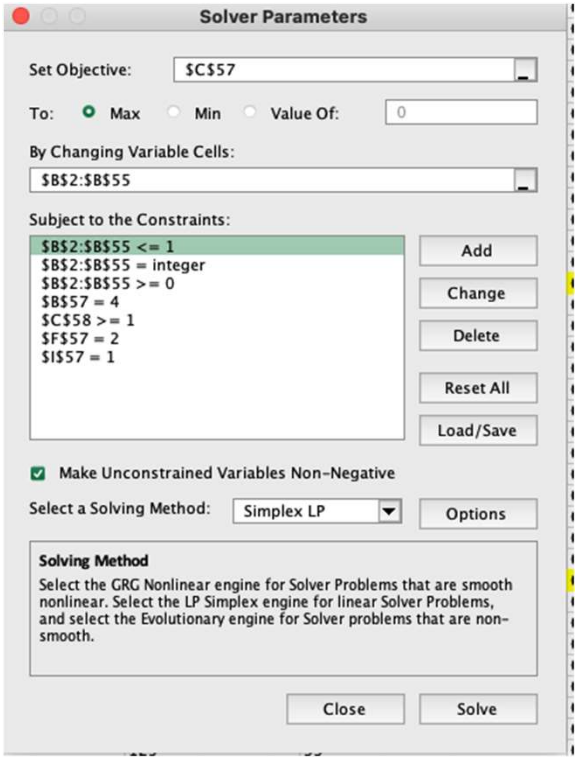
```
puckrecovery_sum = rowSums(Player_merge_final_puckrecovery[2:8],na.rm = TRUE)
Total_game_puckrecovery <- 7
Game_played_puckrecovery <- Total_game - rowSums(is.na(Player_merge_final_puckrecovery))
Player_puckrecovery_list <- cbind(Player_merge_final_puckrecovery,puckrecovery_sum,
                                  Game_played_puckrecovery)
```

```
Player_puckrecovery_list$puckrecovery_per_game <-
  puckrecovery_sum/Game_played_puckrecovery
Final_result_puckrecovery<-Player_puckrecovery_list[order
                                                    decreasing = TRUE),]
```

```
Final_merge1 <- merge(Final_result_pass,Final_result_takeaway,by="Player",all = TRUE)
Final_merge <- merge(Final_merge1,Final_result_puckrecovery,by="Player",all = TRUE)
Result_filter<- Final_merge[,c('Player','pass_sum','successful_pass_sum',
                               'pass_successful_rate',
                               'takeaway_sum','Game_played','takeaway_per_game',
                               'puckrecovery_sum','puckrecovery_per_game')]
median_takeaway_sum <- mean(Result_filter$takeaway_sum,na.rm = TRUE)
average_player <- filter(Result_filter,takeaway_sum>median_takeaway_sum)
average_player
```

Penalty kill players dataset

	A	B	C	D	E	F	G	H	I	J
1	Player	X	takeaway_sum	pass_sum	successful_pass_sum	pass_successful_rate	Game_play	takeaway_per_game	puckrecovery_s	puckrecovery_per_gam
2	Dominique Kremer	1	29	157	99	0.6305732	6	4.833333	134	22.333333
3	Alyson Matteau	0	28	179	130	0.726257	6	4.666667	148	24.666667
4	Breanne Wilson-Bennett	0	25	130	95	0.7307692	6	4.166667	107	17.833333
5	Marie-Jo Pelletier	0	24	185	131	0.7081081	6	4	145	24.166667
6	Natalie Marcuzzi	0	23	102	65	0.6372549	6	3.833333	90	15
7	Shiann Darkangelo	0	23	217	154	0.7096774	6	3.833333	141	23.5
8	Jillian Dempsey	0	22	116	80	0.6896552	6	3.666667	72	12
9	Mallory Souliotis	1	22	259	175	0.6756757	7	3.142857	196	28
10	Jordan Juron	0	21	118	76	0.6440678	6	3.5	95	15.833333
11	Lenka Curmova	0	21	91	50	0.5494505	5	4.2	88	14.666667
12	Sydney Baldwin	0	21	161	112	0.6956522	4	5.25	126	31.5
13	Cassidy MacPherson	0	20	123	83	0.6747967	6	3.333333	92	15.333333
14	Lauren Kelly	0	19	219	142	0.6484018	7	2.714286	174	24.857143
15	Mikyla Grant-Mentis	0	19	148	104	0.7027027	6	3.166667	102	17
16	Stephanie Anderson	0	19	99	69	0.6969697	3	6.333333	60	15
17	Samantha Davis	0	18	167	119	0.7125749	7	2.571429	111	15.857143
18	Kristin Lewicki	0	17	121	82	0.677686	6	2.833333	84	14
19	Lexie Laing	0	17	121	75	0.6198347	6	2.833333	74	10.571429
20	Lindsay Eastwood	0	17	220	150	0.6818182	6	2.833333	152	25.333333
21	Meghara McManus	0	17	59	31	0.5254237	6	2.833333	51	7.285714
22	Shannon Doyle	0	17	158	99	0.6265823	4	4.25	125	31.25
23	Taylor Turnquist	0	17	202	148	0.7326733	6	2.833333	156	22.285714
24	Tereza Vanisova	1	17	173	138	0.7976879	7	2.428571	116	16.571429
25	Kaleigh Fratkin	0	16	332	242	0.7289157	7	2.285714	233	33.285714
26	Maggie LaGue	0	16	132	89	0.6742424	4	4	108	27
27	Hanna Beattie	0	15	95	52	0.5473684	4	3.75	98	24.5
28	Leila Kilduff	0	15	67	47	0.7014925	3	5	40	13.333333
29	Mackenzie MacNeil	0	15	79	47	0.5949367	5	3	78	13
30	Madison Packer	0	15	98	74	0.755102	3	5	56	18.666667
31	Whitney Dove	0	15	156	105	0.6730769	5	3	149	24.833333
32	Allie Thunstrom	0	14	69	45	0.6521739	4	3.5	57	14.25
33	Carlee Turner	0	14	79	57	0.721519	6	2.333333	63	9
34	Christina Putigna	0	14	145	98	0.6758621	6	2.333333	102	14.571429
35	Emma Greco	0	14	126	67	0.531746	5	2.8	117	19.5
36	Erin Gehen	0	14	111	67	0.6036036	6	2.333333	83	13.833333
37	Megan Delay	0	14	86	44	0.5116279	6	2.333333	92	15.333333
38	Taylor Woods	1	14	203	157	0.773399	5	2.8	175	29.166667
39	Tori Sullivan	0	14	102	86	0.8431373	7	2	58	8.285714
40	Emily Janiga	0	13	49	30	0.6122449	3	4.333333	36	12
41	Nina Rodgers	0	13	66	49	0.7424242	4	3.25	48	12
42	Rebecca Morse	0	13	87	63	0.7241379	3	4.333333	72	24
43	Sarah-Eve Coutu Godbou	0	13	94	56	0.5957447	6	2.166667	70	11.666667
44	Tori Howran	0	13	109	74	0.6788991	3	4.333333	90	30
45	Emma Vlasic	0	12	80	58	0.725	4	3	62	15.5
46	Emma Woods	0	12	125	99	0.792	4	3	54	13.5
47	Maddie Rowe	0	12	123	80	0.6504065	4	3	103	25.75
48	Meaghan Rickard	0	12	50	29	0.58	6	2	48	6.857143
49	Katelynn Russ	0	11	70	40	0.5714286	4	2.75	56	14
50	Kayla Friesen	0	11	65	36	0.5538462	3	3.666667	60	20
51	Megan Quinn	0	11	153	107	0.6993464	5	2.2	137	22.833333
52	Meghan Lorence	0	11	102	73	0.7156863	2	5.5	53	13.25
53	Sarah Schwenzfeier	0	11	46	23	0.5	4	2.75	48	12
54	Sarah Steele	0	11	213	158	0.741784	4	2.75	141	23.5
55	Taytum Clairmont	0	11	109	82	0.7522936	4	2.75	67	11.166667
56										
57		4	82			2			1	
58		4	5							



Rotman

Appendix

- Power Play R code:

```
NWHL <- read.csv("/Users/Sameer/Desktop/Rotman MMA Summer Datathon NWHL.csv",header = TRUE)
```

```{r}
#### Power Play Specialists
# find goals in every game
Goal_January23 <- filter(NWHL, Event == "Goal", game_date == "23-01-2021")

Goal_January24 <- filter(NWHL, Event == "Goal", game_date == "24-01-2021")

Goal_January26 <- filter(NWHL, Event == "Goal", game_date == "26-01-2021")

Goal_January27 <- filter(NWHL, Event == "Goal", game_date == "27-01-2021")

Goal_January30 <- filter(NWHL, Event == "Goal", game_date == "30-01-2021")

Goal_January31 <- filter(NWHL, Event == "Goal", game_date == "31-01-2021")

Goal_February01 <- filter(NWHL, Event == "Goal", game_date == "01-02-2021")
```

```{r}
# find players and the number of their goals in every game
Jan23 = Goal_January23 %>% group_by(Player) %>%
                    summarise(total_goal_Jan23 = length(Event))

Jan24 = Goal_January24 %>% group_by(Player) %>%
                    summarise(total_goal_Jan24 = length(Event))

Jan26 = Goal_January26 %>% group_by(Player) %>%
                    summarise(total_goal_Jan26 = length(Event))

Jan27 = Goal_January27 %>% group_by(Player) %>%
                    summarise(total_goal_Jan27 = length(Event))

Jan30 = Goal_January30 %>% group_by(Player) %>%
                    summarise(total_goal_Jan30 = length(Event))

Jan31 = Goal_January31 %>% group_by(Player) %>%
                    summarise(total_goal_Jan31 = length(Event))

Feb01 = Goal_February01 %>% group_by(Player) %>%
                    summarise(total_goal_Feb01 = length(Event))
```

```
### Merging shots and goals
shot_goal_merge <- merge(Final_result_goal, Final_result_Shot , by="Player" , all = TRUE)

shot_goal_merge_final <- filter(shot_goal_merge, goal_sum != "NA")

# Faceoff Specialists

Faceoff_specialists <- filter(NWHL, Event == "Faceoff Win")

Faceoff =  Faceoff_specialists %>% group_by(Player) %>%
                    summarise(Total_faceoff_wins=length(Event))

Faceoff_merge <- merge(shot_goal_merge_final,Faceoff, by="Player", all =TRUE )

Faceoff_merge_final <- filter(Faceoff_merge ,goal_sum != "NA")

# Puck recovery specialists
Puckrecovery_specialists <- filter(NWHL, Event == "Puck Recovery")

Puckrecovery_count = Puckrecovery_specialists %>% group_by(Player) %>%
                    summarise(Puckrecovery=length(Event))

puckrecovery_merge <- merge(Faceoff_merge_final, Puckrecovery_count, by ="Player" , all =TRUE)

puckrecovery_merge_final <- filter(puckrecovery_merge ,goal_sum != "NA")

# pressure metric

Pressure_metric <- filter(NWHL, NWHL$X.Coordinate >= 100)

Pressure_metric_count = Pressure_metric %>% group_by(Player) %>%
                    summarise(Pmetric=length(X.Coordinate)

count(puckrecovery_merge_final)

nrow(puckrecovery_merge_final)


pmetric_merge<- merge(puckrecovery_merge_final, Pressure_metric_count, by = "Player", all =TRUE)

pmetric_merge_final <- filter(pmetric_merge, goal_sum != "NA")


library(writexl)
library(xlsx)

write_xlsx(pmetric_merge_final, '/Users/Sameer/Desktop/lpp.xlsx')
```

```
# calculate the total number of goals for each player
goal_sum = rowSums(Player_goal_merge_final[2:8], na.rm = TRUE)
Total_game <- 7
Games_played <- Total_game - rowSums(is.na(Player_goal_merge_final))
Player_goal_list <- cbind(Player_goal_merge_final, goal_sum, Games_played)
Player_goal_list
```

```{r}
# find each player's number of goals per game

Player_goal_list$goal_per_game <- goal_sum/Games_played
Final_result_goal <- Player_goal_list[order(Player_goal_list$goal_per_game, decreasing = TRUE),]


Final_result_goal <- select(Final_result_goal,Player,goal_sum,Games_played,goal_per_game)
```

```
### Shot Accuracy
# find shots in every game
Total_Shots <- filter(NWHL, Event == "Shot")


# find players and the number of their shots in every game
shot_table= Total_Shots %>% group_by(Player) %>%
                    summarise(Total_Shots = length(Event))


# Shot detail
Succ_Shot <- filter(NWHL, Event == "Shot" & Detail.2 == "On Net")

Unsucc_Shot <- filter(NWHL, Event == "Shot" & (Detail.2 == "Missed" | Detail.2 == "Blocked"))

Succ_Shot = Succ_Shot%>% group_by(Player) %>%
                    summarise(Successful = length(Event))

Unsucc_Shot = Unsucc_Shot%>% group_by(Player) %>%
                    summarise(Unsuccessful = length(Event))

names(Succ_Shot)[names(Succ_Shot) == "Total_Shots"] <- "Successful"
names(Unsucc_Shot)[names(Unsucc_Shot) == "Total_Shots"] <- "Unsuccessful"

shots <- distinct(shot_table)

Player_shot_merge <- merge(shot_table, Succ_Shot, by = "Player")

Player_shot_merge <- within(Player_shot_merge, Accuracy<-(Successful/Total_Shots)*100)

Final_result_Shot <- Player_shot_merge[order(Player_shot_merge$Accuracy, decreasing = TRUE),]

Final_result_shot
```

Rotman

Player	X	goal_sum	Games_played	goal_per_game	Total_Shots	Successful	Accuracy	Total_faceoff_wins	uckrecover	Pmetric
Mikyla Grant-Mentis	1	5	4	1.25	43	26	53.061224	19	102	318
Samantha Davis	1	4	3	1.333333333	34	19	55.882353	5	111	328
Taylor Woods	1	4	4	1	42	10	23.809524	2	175	247
Autumn MacDougall	0	3	2	1.5	18	11	61.111111	0	77	190
Jillian Dempsey	1	3	2	1.5	24	14	58.333333	70	72	234
Mallory Souliotis	0	3	2	1.5	32	14	43.75	0	196	230
McKenna Brand	1	3	3	1	55	23	41.818182	8	111	371
Audra Richards	0	2	2	1	14	9	64.285714	2	51	97
Breanne Wilson-Bennet	0	2	1	2	31	17	54.83871	45	107	280
Brooke Boquist	0	2	2	1	27	14	51.851852	2	48	168
Christina Putigna	0	2	2	1	37	21	56.756757	3	102	278
Haley Mack	0	2	2	1	13	9	69.230769	6	59	163
Katelynn Russ	0	2	2	1	26	21	80.769231	6	56	148
Kristin Levicki	0	2	2	1	21	10	47.619048	1	84	212
Lauren Kelly	0	2	2	1	42	14	33.333333	0	174	227
Leila Kilduff	0	2	1	2	5	2	40	0	40	38
Mackenzie MacNeil	0	2	1	2	26	16	61.538462	1	78	212
Meaghan Rickard	0	2	1	2	18	8	44.444444	0	48	127
Nina Rodgers	0	2	1	2	17	5	29.411765	1	48	104
Alyssa Wohlfeiler	0	1	1	1	24	12	50	0	52	175
Amanda Conway	0	1	1	1	10	5	50	0	24	57
Amy Curlew	0	1	1	1	9	5	55.555556	4	32	90
Cailey Hutchison	0	1	1	1	4	2	50	18	21	43
Emily Fluke	0	1	1	1	11	3	27.272727	4	44	106
Emily Janiga	0	1	1	1	8	4	50	17	36	82
Emma Vlasic	0	1	1	1	15	8	53.333333	45	62	138
Haylea Schmid	0	1	1	1	9	5	55.555556	11	27	77
Jonna Curtis	0	1	1	1	20	15	75	26	72	181
Jordan Juron	0	1	1	1	29	18	62.068966	49	95	238
Kaycie Anderson	0	1	1	1	14	5	35.714286	1	38	70
Kayla Friesen	0	1	1	1	23	11	47.826087	31	60	134
Lexie Laing	0	1	1	1	23	13	56.521739	55	74	232
Lindsay Eastwood	0	1	1	1	36	8	22.222222	0	152	198
Lisa Chesson	0	1	1	1	9	1	11.111111	0	62	54
Mallory Rushton	0	1	1	1	10	6	60	1	30	75
Meaghan Pezon	0	1	1	1	8	2	25	2	44	93
Megan Quinn	0	1	1	1	15	5	33.333333	0	137	124
Meghara McManus	0	1	1	1	24	15	62.5	2	51	150
Rebecca Russo	0	1	1	1	7	4	57.142857	2	45	86
Sarah-Eve Coutu Godb	0	1	1	1	42	26	61.904762	2	70	210
Sydney Baldwin	0	1	1	1	32	13	40.625	0	126	136
Taylor Wenczkowski	0	1	1	1	36	17	47.222222	4	90	216
Taytum Clairmont	0	1	1	1	13	7	53.846154	18	67	193
Theresa Knutson	0	1	1	1	16	9	56.25	2	32	86
Tori Howran	0	1	1	1	13	5	38.461538	0	90	100
Winny Brodt-Brown	0	1	1	1	0	0	0	0	48	33
	5	19						1	1	1
	5							1	1	1
		5								

Solver Parameters

Set Objective: $C$50

To: Max Min Value Of: 5

By Changing Variable Cells:

$B$2:$B$47

Subject to the Constraints:

$B$2:$B$47 <= 1
$B$2:$B$47 = integer
$B$2:$B$47 >= 0
$B$50 = 5
$C$53 >= 3
$I$50 = 1
$J$50 = 1
$K$50 = 1

Add
Change
Delete
Reset All
Load/Save

Make Unconstrained Variables Non-Negative

Select a Solving Method: Simplex LP

Options

Solving Method

Select the GRG Nonlinear engine for Solver Problems that are smooth nonlinear. Select the LP Simplex engine for linear Solver Problems, and select the Evolutionary engine for Solver problems that are non-smooth.

Rotman