Solution - Big Buck Challenge by IEOR@IITB and McKinsey Knowledge Center India

Presented by
- Aakash Kerawat
Final year UG Student
IIT Roorkee

Outline

1. Predicting Withdrawal
   a. First thought
   b. Data preprocessing
   c. EDA
   d. Feature Engineering
   e. Model Selection
2. Optimizing replenishment
   a. Cost function
   b. Optimization
   c. Plots

Predicting Withdrawal

First Thought

Looking at the data there can be two approaches -

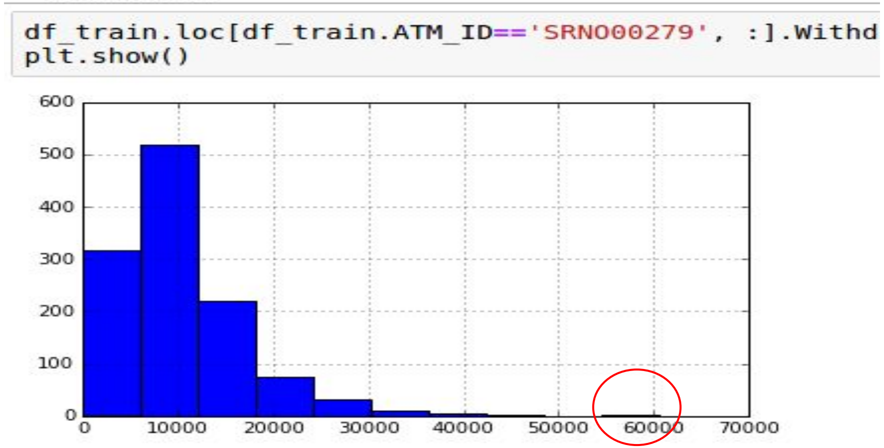
1. Time series forecasting
2. Using machine learning models.

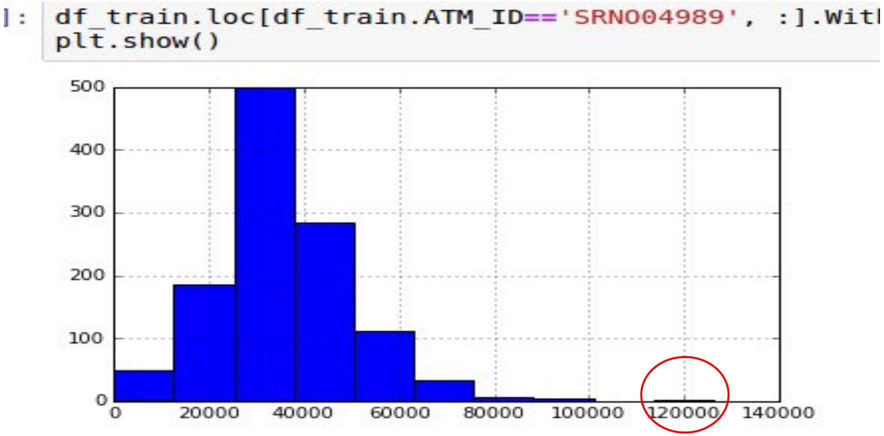
Chose option 2 as -

A. There was missing data.
B. ML models can perform at par (or better) than TS models with right feature set.

Data Preprocessing

- Distribution of ATM_ID specific withdrawal indicated high-value outliers.
- Removed the data points with withdrawal value > 99 percentile of that ATM_ID..

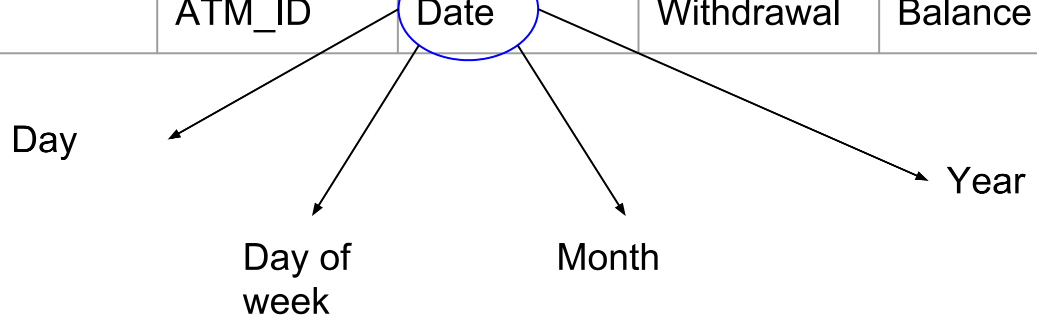




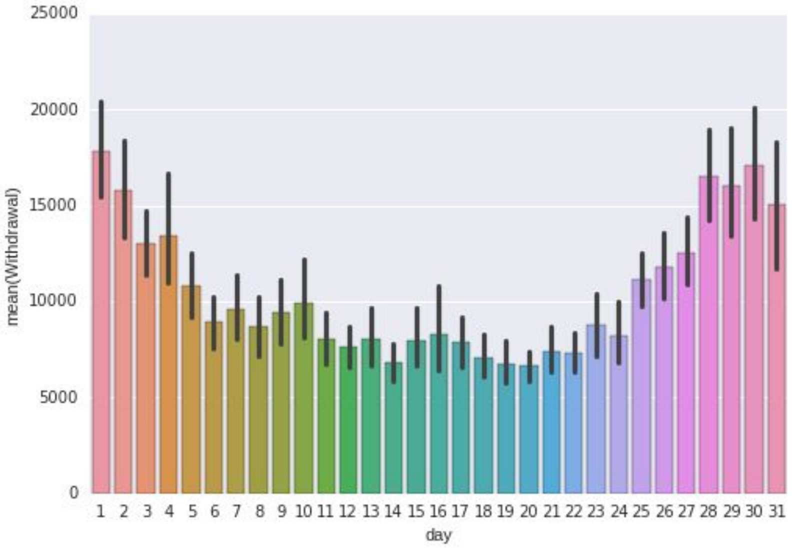
EDA

- The given data -
-

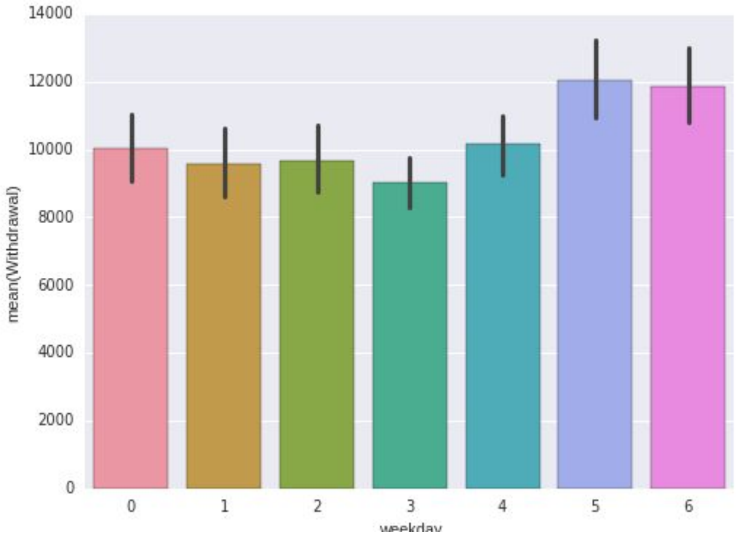
ID	ATM_ID	Date	Withdrawal	Balance



```
sns.barplot(df_train.loc[df_train.ATM_ID=='SRN000279', 'day'],
plt.show()
```



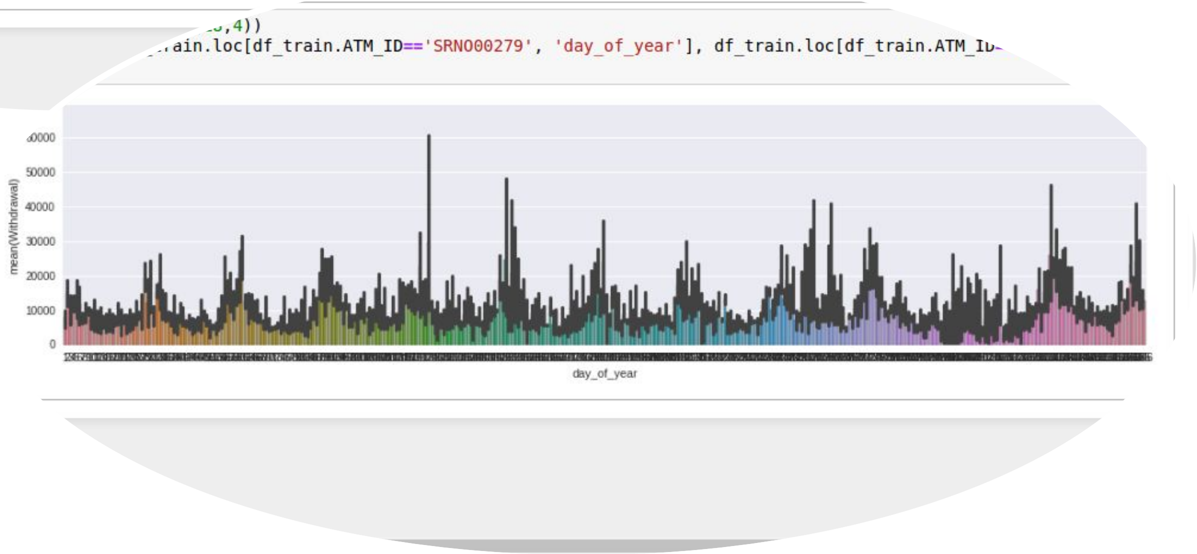

```
sns.barplot(df_train.loc[df_train.ATM_ID=='SRN000279', 'weekday'],
plt.show()
```



Trends

Trends


_train.loc[df_train.ATM_ID=='SRN000279',
14000
12000
10000
8000
6000
4000
2000
0
mean(Withdrawal)
1
2
3
4
5
6
7
8
9
10
11
12
month


,4))
ain.loc[df_train.ATM_ID=='SRN000279', 'day_of_year'], df_train.loc[df_train.ATM_ID
50000
40000
30000
20000
10000
0
mean(Withdrawal)
day_of_year

Final Date features

- Day
- Day of week
- Month
- Day of year
- ~~Year~~
    - Removed as test data was only of 1 particular year.
    - Also there wasn't a significant trend.

Additional Features

Merged the ATM_info data which provided -

- Type
- Facility

Engineered Features

Motivation - Initially thought of making different model for each ATM, but this has many limitations.

Following features were added -

- Withdrawal_mean - Mean value of withdrawal amount for a particular ATM_ID
- Withdrawal_std - Std. deviation value of withdrawal amount for a particular ATM_ID

Engineered Features

These features gave a sense of distribution of withdrawal for that ATM to the model as a continuous value feature.

- Withdrawal_uq - Upper quartile of withdrawal amount for a particular ATM_ID
- Withdrawal_lq - Lower quartile value of withdrawal amount for a particular ATM_ID

Model Selection

- Tried a range of algorithms from Linear regression, Decision Tree regressor and Random Forest regressor to Gradient Boosting regressor.
- Chose Gradient boosting as the final model.

Reasons

- All the date features can be considered as categorical and one-hot encoding would result in more than 300 features.
- Data had both categorical and continuous features.
- Tree based algorithms are good at handling both at the same time.

Reasons

- Tree based algorithms can learn from categorical features without one-hot encoding.
- **Another major reason** - Checked the cross-validation score for all the models and gradient boosting came out to be the best.

Optimizing Replenishment

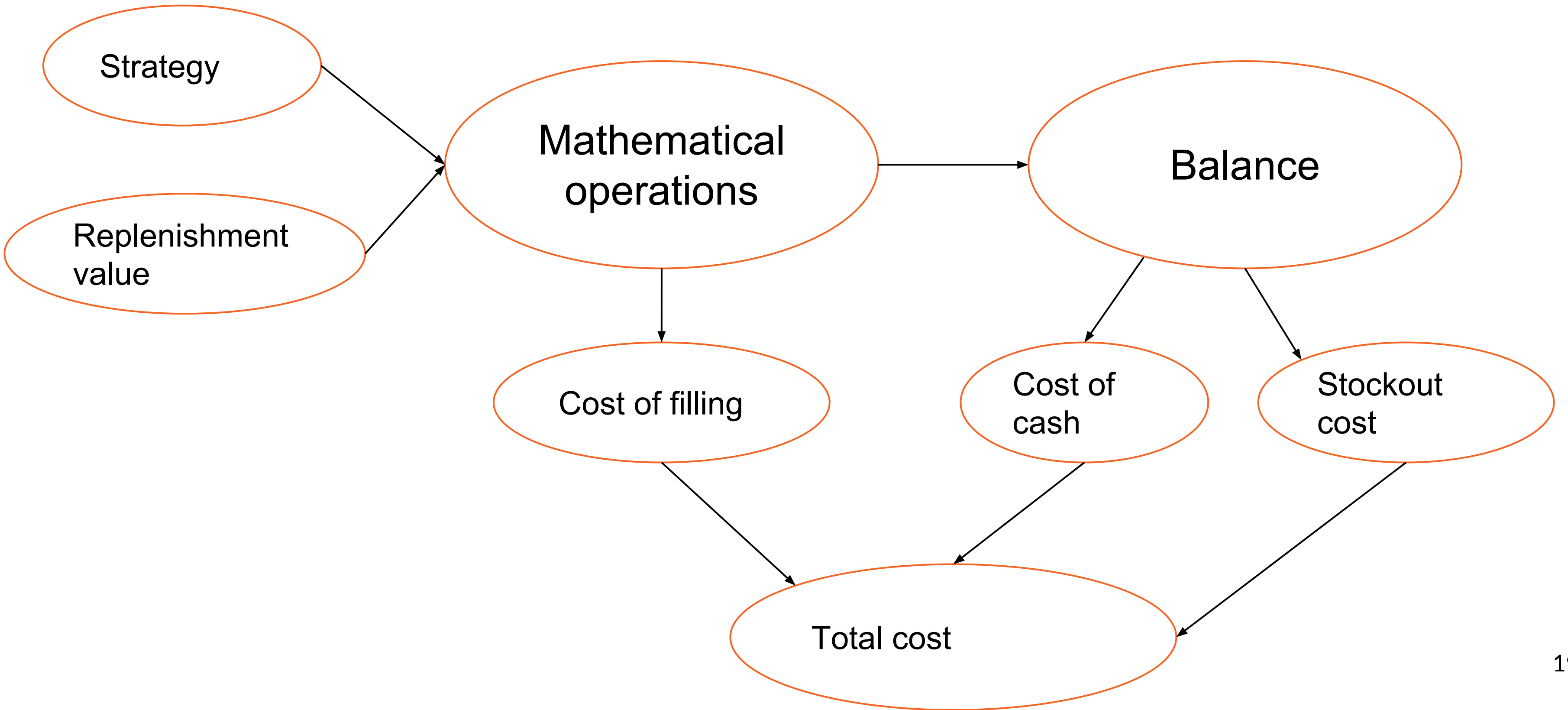
Approach

Calculated the total cost for each strategy for a range of replenishment values and finally chose the pair for which the cost was minimum.

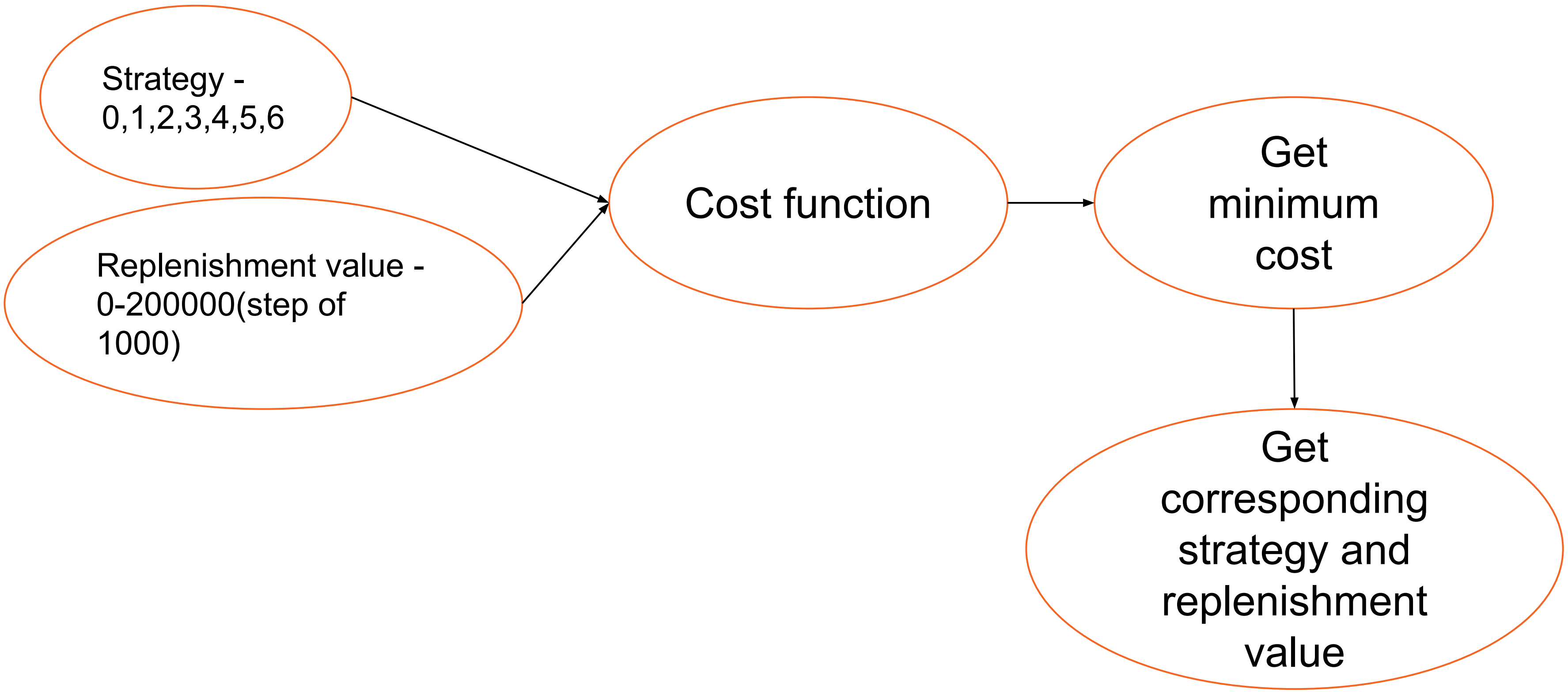
Defined 2 functions

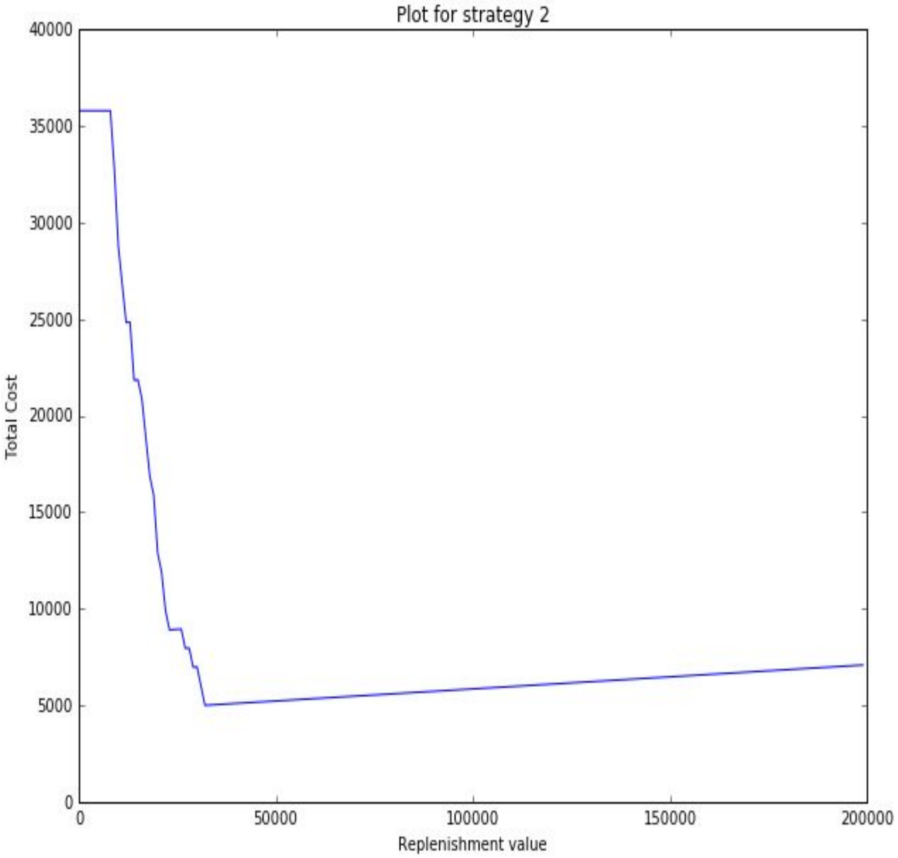
- Cost function
- Optimization function

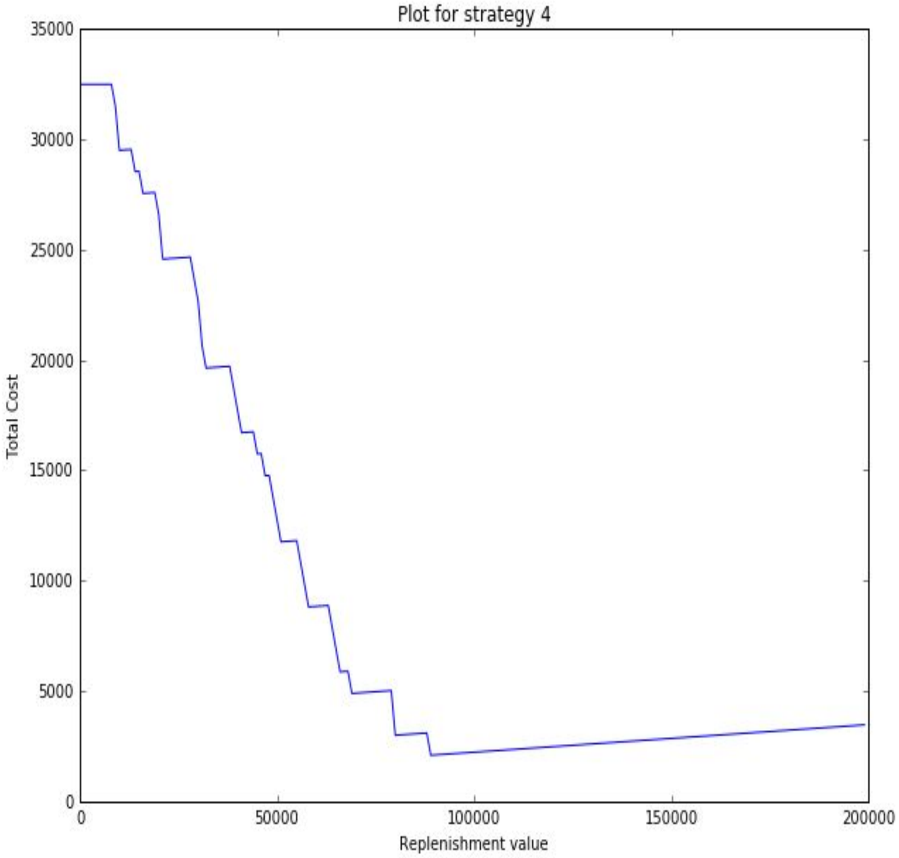
Cost function

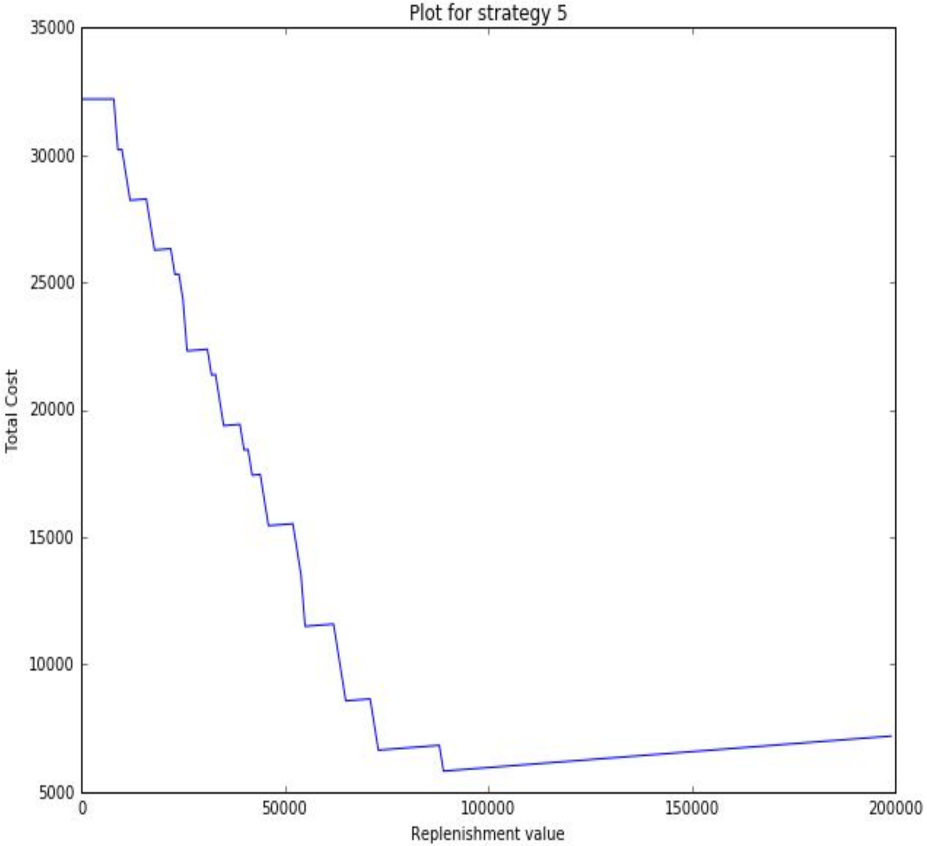


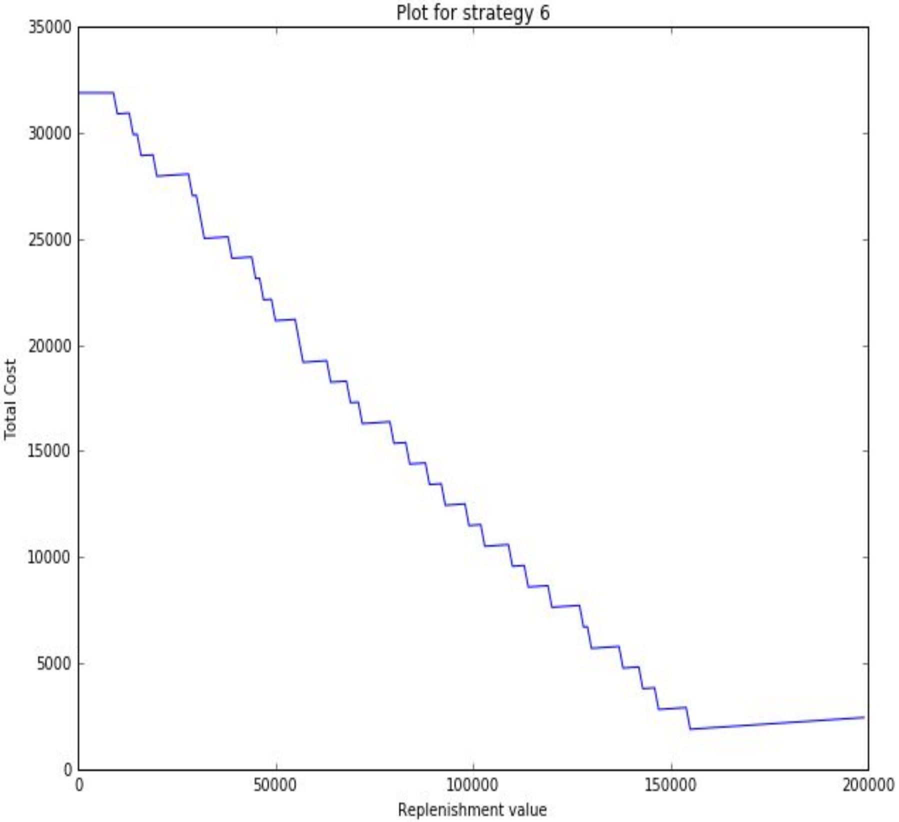
Optimize


Strategy -
0,1,2,3,4,5,6
Replenishment value -
0-200000(step of
1000)
Cost function
Get
minimum
cost
Get
corresponding
strategy and
replenishment
value


Plot for strategy 2
Total Cost
Replenishment value
0
5000
10000
15000
20000
25000
30000
35000
40000
50000
100000
150000
200000


Plot for strategy 4
Total Cost
Replenishment value
0
5000
10000
15000
20000
25000
30000
35000
50000
100000
150000
200000


Plot for strategy 5
Total Cost
Replenishment value
35000
30000
25000
20000
15000
10000
5000
0
50000
100000
150000
200000


Plot for strategy 6
Total Cost
Replenishment value
35000
30000
25000
20000
15000
10000
5000
0
0
50000
100000
150000
200000

Thank You