Grouping of Non-Daily Trains into Daily Trains

Abijith P.Y,Sajith Menon,Zubeen Kishore Borkar

***Abstract*—Golden Quadrilateral route accounts for more than 70% of traffic on the Indian railway, so it's important to use this network efficiently. In this project, we are considering two routes of a Quadrilateral network viz MMCT-NDLS and MAS-NDLS. We used class-based implementation with Train class having different attributes such as arrival time, departure time, station visited, etc. K Means Clustering algorithm was chosen to cluster train into groups with similar property. We compared this cluster to find a maximum dailyzing schedule. Dialyzing term will often be used in this report which means to convert a nondaily train running in a certain path to a daily or near-daily train with similar coverage in terms of the path covered. Clustering results produced can help the Indian railway to make the right decision on adding or removing certain trains in a route. This decision can help them to optimize their profit and people can also get an efficient service with less overall delay..**

***Index Terms*—Dailyzing, Clustering, Quadrilateral, non-daily trains .**

I. Introduction

What is the problem ?

The Problem we are primarily working on in this project is optimizing the timetable of the train such that the golden quadrilateral route is efficiently used. We were given trains schedule for 2 routes: MAS-NDLS and MMCT-NDLS with up and down classification. We needed to identify daily and non daily trains from the existing schedule and had to cluster them into some groups based on various constraints like time slots, arrival time, traveling days of trains etc.

Why is it important to solve it?

In the Indian railway network, Golden Quadrilateral routes consist of Chennai, Mumbai, Delhi and Kolkata account for a major portion of the traffic. So it is necessary to optimize the trains schedule in this route as it gives maximum profit for the Indian railway. By reducing wastage of time in the network people also could get benefitted from getting better connectivity across different metropolitan cities.

What challenges one faces while solving the problem ?

The Biggest challenge was found in forming a dailyzing path. Dialyzing is a term that refers to converting a group of non-daily trains into a daily train schedule. For example, there exists a train from Delhi which runs on Monday Tuesday Wednesday at 9 am. Our problem required us to group the trains which run on similar paths with similar time on Thursday, Friday, Saturday, Sunday or the maximum possible day to form a single group. Choosing the best criteria was difficult. Some trains only traveled one or two stations in our route. Finding an optimal parameter set for grouping is the biggest challenge in solving this problem.

Salient features of your work

A salient feature of our work is we were able to give a clustering based on a non-daily train running on complimentary days with maximum network usage. We used a class-based implementation and our work will be applicable for new datasets(routes) as well. We don't have to tweak our code to run for finding clusters in new routes. Our result grouped trains with maximum coverage of weekdays thus making it an optimal dailyzing path.

Summary of results:

Our final result has clusters of trains with similar properties but complemented weekdays. We initially clustered trains into groups using the k-means technique. Our clustering criteria was basically distance covered by train. We then compared this result with trains running on complimentary days and grouped trains with maximum 1's that is it will run for most of the day on the same path, in essence, we are finding an optimal dailyzing train clustering.

II. Background and Previous Work

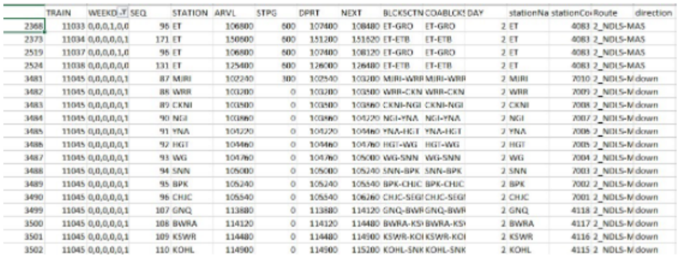
Background: Optimization of Train schedule is one of the projects undertaken by IIT Bombay. Dailyzing train schedule, a submodule we worked on in this project is a part of a project "optimal train scheduling".This project is Supervised by Prof Madhu Belur and prof Narayan Rangaraj.A lot of previous work has done in this project by many people across different departments of IIT Bombay. Some of the people who worked on this project shared their experiences and clarified our doubts before initiating this project.

Previous work: The team which initially worked on this project tried to achieve the objective using a manual method in which they used Excel as their tool to work on it. The previous method was cumbersome and had some human errors. Also in case, they had to work on a new route, the process had to be repeated again which is not an efficient way of doing work. Then this project was handed over to our team in which we tried to make the problem generalized so that changing the dataset does not affect the performance of the algorithm. We tried to achieve our objective using two approaches. In approach 1, we tried to work on the dataset given without changing it much, more detailed explanation has been given later. In approach 2, we tried the class implementation of the dataset which has been explained later.

III. Datasets

Source of data set - The given data set has been provided to us by Indian Railways which have been modified according to our need.

As observed the dataset gives us information about:- Train number Weekdays (the days on which it travels) List of stations visited by a train Arrival and Departure time of train (given in seconds) The sequence in which the stations are visited by train (BLCKSCTN) Current Station Code Route on which the train travels Direction (UP and DOWN)



	TRAIN	WEEKD	SEQ	STATION	ARVL	STPG	DPRT	NEXT	BLCKSCTN	COABLCKS	DAY	stationNa	stationCo	Route	direction
2368	11033	0,0,0,1,0,0	96	ET	106800	600	107400	108480	ET-GRO	ET-GRO	2	ET	4083	2_NDLS-MAS	
2373	11034	0,0,0,0,0,1	171	ET	150600	600	151200	151620	ET-ETB	ET-ETB	2	ET	4083	2_NDLS-MAS	
2519	11037	0,0,0,0,1,0	96	ET	106800	600	107400	108120	ET-GRO	ET-GRO	2	ET	4083	2_NDLS-MAS	
2524	11038	0,0,0,0,0,0	131	ET	125400	600	126000	126480	ET-ETB	ET-ETB	2	ET	4083	2_NDLS-MAS	
3481	11045	0,0,0,0,0,1	87	MJRI	102240	300	102540	103200	MJRI-WRR	MJRI-WRR	2	MJRI	7010	2_NDLS-M	down
3482	11045	0,0,0,0,0,1	88	WRR	103200	0	103200	103500	WRR-CKN	WRR-CKN	2	WRR	7009	2_NDLS-M	down
3483	11045	0,0,0,0,0,1	89	CKNI	103500	0	103500	103860	CKNI-NGI	CKNI-NGI	2	CKNI	7008	2_NDLS-M	down
3484	11045	0,0,0,0,0,1	90	NGI	103860	0	103860	104220	NGI-YNA	NGI-YNA	2	NGI	7007	2_NDLS-M	down
3485	11045	0,0,0,0,0,1	91	YNA	104220	0	104220	104460	YNA-HGT	YNA-HGT	2	YNA	7006	2_NDLS-M	down
3486	11045	0,0,0,0,0,1	92	HGT	104460	0	104460	104760	HGT-WG	HGT-WG	2	HGT	7005	2_NDLS-M	down
3487	11045	0,0,0,0,0,1	93	WG	104760	0	104760	105000	WG-SNN	WG-SNN	2	WG	7004	2_NDLS-M	down
3488	11045	0,0,0,0,0,1	94	SNN	105000	0	105000	105240	SNN-BPK	SNN-BPK	2	SNN	7003	2_NDLS-M	down
3489	11045	0,0,0,0,0,1	95	BPK	105240	0	105240	105540	BPK-CHJC	BPK-CHJC	2	BPK	7002	2_NDLS-M	down
3490	11045	0,0,0,0,0,1	96	CHJC	105540	0	105540	106260	CHJC-SEGI	CHJC-SEGI	2	CHJC	7001	2_NDLS-M	down
3499	11045	0,0,0,0,0,1	107	GNQ	113880	0	113880	114120	GNQ-BWR	GNQ-BWR	2	GNQ	4118	2_NDLS-M	down
3500	11045	0,0,0,0,0,1	108	BWRA	114120	0	114120	114480	BWRA-KS	BWRA-KS	2	BWRA	4117	2_NDLS-M	down
3501	11045	0,0,0,0,0,1	109	KSWR	114480	0	114480	114900	KSWR-KOI	KSWR-KOI	2	KSWR	4116	2_NDLS-M	down
3502	11045	0,0,0,0,0,1	110	KOHL	114900	0	114900	115200	KOHL-SNK	KOHL-SNK	2	KOHL	4115	2_NDLS-M	down

Fig. 1. Crude Dataset Provided by Indian Railway

Challenges in assembling the data :

The data provided to us is given in a crude form and it contains a lot of irrelevant information so we dropped those features which were not useful to us. This is a necessary step because irrelevant data can affect the performance of the clustering algorithm. There were some trains which visited just a single station, such trains should be clustered in a single cluster (these trains are not useful for us according to our objective) We had to separate daily and non-daily trains since our work focused on non-daily trains. We separated our data set based on routes and then they were again separated based on direction (UP or DOWN)

We tried to solve our problem using the same data set in two different ways which are discussed in procedure and experiments topic.

IV. Procedure and Experiments :

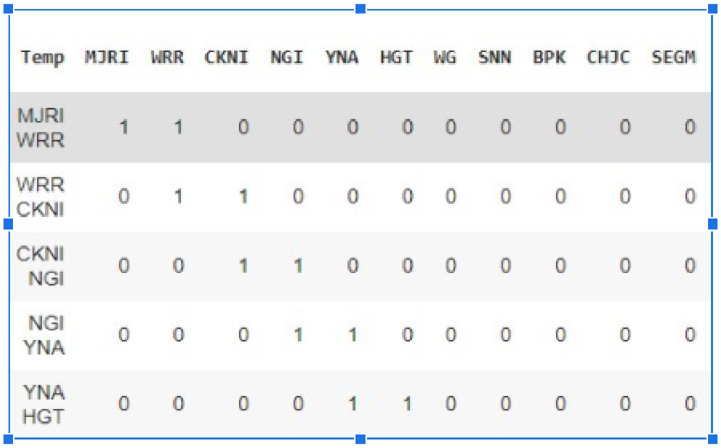
Libraries used : Pandas - for creating Database Numpy - for working with arrays Matplotlib - for plotting Sklearn - consist of efficient tools for machine learning Kneed - for knee location in elbow method

We processed our dataset in which we removed columns that were irrelevant to us. Then we separated daily and non-daily trains as our work focused on non-daily trains clustering. The further classification was done based on direction (UP and DOWN) An observation was made that the COABLCKSCTN column helps us to know the sequence of the next station visited by train, so this was used as the prime feature to try to cluster the data. But since direct usage of the feature was impossible, we separated it into unique stations and then assigned values (as 1) to those stations which are being visited and (0) to those stations which are not being visited.

Example : We have 5 stations namely A, B, C, D, E, and two trains name train 1 and train 2. Let train 1 travels from A-B-C and train 2 travels from C-D-E so according to our logic we will show them as

Train number	COABLCKSCTN	A	B	C	D	E
Train1	A-B	1	1	0	0	0
Train1	B-C	0	1	1	0	0
Train2	C-D	0	0	1	1	0
Train2	D-E	0	0	0	1	1

Table 1 : Encoding Station



Temp	MJRI	WRR	CKNI	NGI	YNA	HGT	WG	SNN	BPK	CHJC	SEGM
MJRI WRR	1	1	0	0	0	0	0	0	0	0	0
WRR CKNI	0	1	1	0	0	0	0	0	0	0	0
CKNI NGI	0	0	1	1	0	0	0	0	0	0	0
NGI YNA	0	0	0	1	1	0	0	0	0	0	0
YNA HGT	0	0	0	0	1	1	0	0	0	0	0

Fig. 2. Encoding Sation w.r.t Station Travelled

Doing so helps us to get an idea of the sequence of stations visited by trains.

After doing basic coding, our data set looks like the example we demonstrated. Now, we keep only that column that has the station's name. These columns will be provided as input to the elbow method and silhouette score method to find the optimal number of clusters.

The results obtained from both silhouette score and elbow method were not satisfactory as they couldn't give us the optimal number of clusters required for further processing. This indicates that the logic used by us has some flaws in it. So to test this approach, we took the optimal number of clusters as 20, applied the K-means method, and tried to fit the model accordingly.

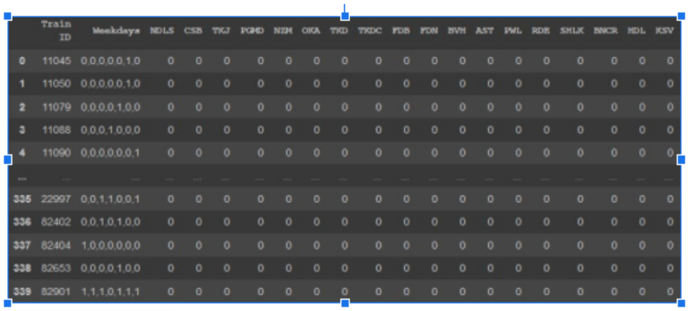
The results obtained were poor (CSV file has been provided) and thus this logic was discarded by us as it didn't give us the results that were required.

Approach 2 : (Class implementation)

After training the model in Approach 1 we realised a huge drawback: the given data is made such that there are multiple entries for the same train which was skewing the clusters. Therefore we decided to generate specialized objects called "Train" which stored the necessary data required for each train id as it's properties viz. We have specifically defined the given properties: Start station, end station, stations involved in it's journey, weekdays it is travelling and direction it is travelling ; all of this enumerated using their specific train id. These data was stored in a list of objects . By doing this the entire dataset got reduced to very less yet unique data points, as follows:

This dataset was clustered using the same methods used as in Approach 1. Sensible clusters were obtained which were used for further processing. The clusters formed contained trains which had similar journeys or journeys close to similar ones with at least 70% similarity.

After proper clustering is obtained, our task is to make pair of trains (from the same cluster) running on complimentary days of weeks. To get a better understanding, let's consider an example given below.



	Train ID	Weekdays	NDLS	CSB	TKJ	PGMD	NZM	OKA	TKD	TKDC	FDB	FDN	BVH	AST	PWL	RDE	SHLK	BNCR	HDL	KSV
0	11045	0,0,0,0,0,1,0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	11050	0,0,0,0,0,1,0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	11079	0,0,0,0,1,0,0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	11088	0,0,0,1,0,0,0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	11090	0,0,0,0,0,0,1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
335	22997	0,0,1,1,0,0,1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
336	82402	0,0,1,0,1,0,0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
337	82404	1,0,0,0,0,0,0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
338	82653	0,0,0,0,1,0,0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
339	82901	1,1,1,0,1,1,1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fig. 3. Dataset Representation with Class based Implementation

Let the weekdays - Monday, Tue, ….., Sunday be the order we are considering. Now if a train runs on a given weekday, we mark it as 1 or else 0.

Train number	Weekdays	WeekdaysCompliment
TrainA	1100000	0011111
TrainB	0010000	1101111
TrainC	0001110	1110001

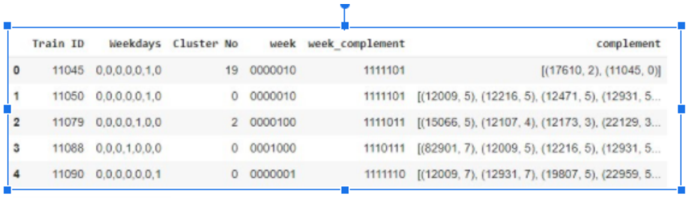
Table 2 : Complimentary Weekday Example

Suppose the above trains belong to the same cluster, now we want the maximum utilization of route, so we make pairs of trains such that they run on different days but their departure timing should have a difference of 20 minutes at max (we need to find out such combinations). Now from the above example, it can be observed that if we make A and B as a pair, apart from the first three days, the remaining days, the route remains unutilized but if we make A and C as a pair, their combination utilizes the route better, so our preferred choice must be to select A and C as a pair. To implement this logic, we use the following algorithm whose logic is explained below

Train_no	Weekdays	Compliment	Combinations
TrainA	1100000	0011111	(B,3),(C,5)
TrainB	0010000	1101111	(A,3),(C,4)
TrainC	0001110	1110001	(A,5),(B,4)

Table 3 : Maximizing Complimentary Weekday Grouping

We try to compare Train A's weekday compliment column with the weekdays of other trains. After comparing, we write the number of elements matched in the combination column. This algorithm gives us the correct combination of train pairs A and C (as C has 5 elements). After applying the given algorithm our data set looks like the figure given below



	Train ID	Weekdays	Cluster No	week	week_complement	complement
0	11045	0,0,0,0,0,1,0	19	0000010	1111101	[(17610, 2), (11045, 0)]
1	11050	0,0,0,0,0,1,0	0	0000010	1111101	[(12009, 5), (12216, 5), (12471, 5), (12931, 5...
2	11079	0,0,0,0,1,0,0	2	0000100	1111011	[(15066, 5), (12107, 4), (12173, 3), (22129, 3...
3	11088	0,0,0,1,0,0,0	0	0001000	1110111	[(82901, 7), (12009, 5), (12216, 5), (12931, 5...
4	11090	0,0,0,0,0,0,1	0	0000001	1111110	[(12009, 7), (12931, 7), (19807, 5), (22959, 5...

Fig. 4. Representation of Weekday Complimentary with Maximum grouping

V. Results

Initial Clusters were formed using an unsupervised learning algorithm viz. K-Means algorithm. The clusters thus formed are further clustered using a custom algorithm made by us. These clusters are formed such that trains with around 70% similarity , as well as those that aren't travelling on the same date. Also,an assumption is made that a route is said to be fully utilized when the time between 2 trains is less than or equal to 20 minutes.

The gaps in the daily schedule are easily visible by plotting each cluster as a function of days of a week. We can easily detect days in which the route is free and can optimize the timetable based on that.

The clustering method proposed in this project is unbiased, automatic, simple, and flexible. Previous methods proposed extracted information mostly through observations, and through statistical analysis and ranking. In most of the methods the entire dataset is used to find trains running on complementary days but with journeys with a similarity of almost 70%. This was highly inefficient as it required almost a day to complete the classification.

Later a Decision Tree model was built but this wasn't able to cluster similar journeys and find journeys on complementary days. Moreover, algorithms to form clusters have to be defined by the user and it wasn't able to work through variability.

The new unsupervised learning based clustering algorithm is able to achieve these objectives and is robust enough to account for variability in a given route data.

Compared to the previous method this method relies on readily available data and does not need detailed knowledge on the routes. It can therefore be scaled to different levels of detail or transferred to other modes of transportation where routes can be defined over a given path, such as bus networks or air traffic.

VI. Conclusion

In this paper, a new method is presented to analyze railway operations, based on unsupervised learning techniques. Previous studies showed that it was necessary for tools to analyse the operation of the railways using data from automatic data sources and detect less used routes automatically. Here we applied K-means clustering to train journey data to identify clusters with similar journey characteristics and utilization of routes. This method is automatic, unbiased, and simple.

The algorithm used by us provides a tool to identify empty routes and optimise their utilization better. The effectiveness of this approach was verified by the dowards train data from the data files. The strength of this approach lies in its simplicity, in addition to flexibility. Unsupervised Learning methods like clustering allow us to identify the internal structures of a system, in contrast with supervised systems that attempt prediction of results on the basis of assumed connections in the input. Therefore methods like Bayesian networks, Support Vector Regression and Neural Network methods require initial assumptions on significant factors that have direct effect on the required output. This might get cumbersome to identify and may be hidden. At times wrong assumptions will lead to faulty and erroneous outputs which may lead to completely wrong conclusions from the data. Here we particularly used the k-means algorithm, which doesn't require initial assumptions of any kind, thus almost all recurrent patterns can be identified. K-means was selected because it is the most common algorithm for partitional clustering. Although in literature there are several classification methods and algorithms, neither is clearly preferable to the rest. It is necessary to note the fact that output of clustering algorithms suggests only the hypotheses and it is the interpretation of results that plays the vital role rather than seeking the best clustering method. Being said that, an improvement may be seen on further research on choice of clustering statistic and also on the choice of the parameters might be supported by advanced techniques and metrics.

Neural network based feature extraction than a manual method can improve the efficiency of the algorithm even more as this would lead to further reduction of biases. An RL based approach can also be implemented along with it which increases the usefulness of clusters even more . The trains that are on the same day but on non intersecting time frames can be clustered together which isn't considered in our clustering methods. This will in turn create better clusters than the current ones and could further enhance by identifying new clustering metrics between observations or integrate additional sources of information to improve cluster inference.

VII. Statement of Contributions

Statement of contributions : Team members for this project are :

Abijith PY (203190024)
Sajith Menon (203190025)
Zubeen Kishore (203190026)

We split the work amongst us as follows:

Literature review - Abijith, Sajith, and Zubeen
Work on Dataset of method 1 - Sajith and Zubeen
Coding for method 1 - Sajith and Zubeen
Work on Dataset of method 2 - Abijith
Coding for method 2 - Abijith
Weekdays algorithm coding - Sajith and Zubeen

Report work was split as follows:

Abstract, Introduction, and Background work - Sajith
Previous work, Dataset, Procedure, and experiments (Method 1 and weekdays algorithm) - Zubeen
Procedure, and experiments (Method 2), Results and conclusion - Abijith
Report formatting in IEEE format - Sajith

PPT work was split as follows:

Introduction and K-means - Sajith
Implementation (approach 1 and weekdays algorithm) - Zubeen
Implementation (approach 2), Results and Future work - Abijith

Video making - Zubeen