Project Report

VLBA II – System Architectures

Project: Maintenance and Services

Team Members

Siddharth Gupta - 241257

Rashid Saeed - 231682

Akshay Jain - 241265

Saikat Mitra - 241248

Helbi Mathew - 244032

Magdeburg, 3. Jul 2023

Otto-von-Guericke-Universität Magdeburg
Magdeburg Research and Competence Cluster
Fakultät für Informatik
Institut für Technische und Betriebliche Informationssysteme
Arbeitsgruppe Wirtschaftsinformatik

Table of contents

Table of figures

Table of tables

1. Introduction

Mobility Worldwide, situated in Magdeburg, is a globally operating corporation specialized in innovation, production, talent promotion, and services in the mobility domain. One of its many departments, Maintenance & Service (MS) is mainly responsible for the functioning of the physical facilities at the client's site.

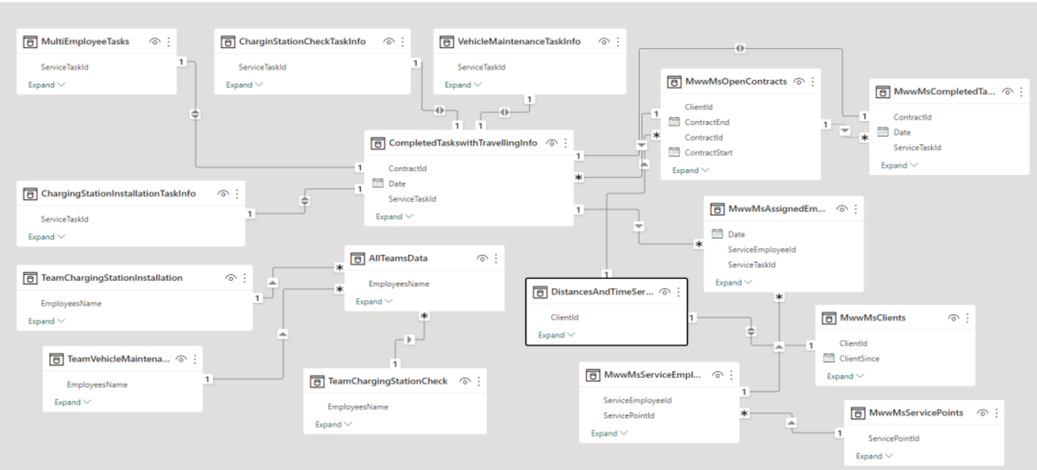
This project aims to improve the quality of services at MWW by optimizing employee assignments and leveraging the expected learning trajectories of individuals employees and employee service teams, who work on variety of services, ranging from installing new charging stations for electric cars to maintaining broken electric bike motors for major customers like the postal service. Business data extracted from the internal SAP DWC system of the company are used for analysis and visualization of the relationships between service employees, their assigned tasks, and the evolution of their learning capabilities.

The historical data available for analysis includes data related to service employees involved, customers for which MWW promises responsibility, open maintenance contracts and service orders performed in the past. Through detailed scrutiny of this data, we aim to identify patterns and extract meaningful insights regarding employee performance and task outcomes. This analysis enables us to identify the best-performing individuals and teams based on their past service records.

We utilize this analysis and visualization data to strategically assign the most qualified individuals to new service requests and most efficient teams for tasks that require more than one person. By forming efficient employee pairs based on their complementary skills and expertise, the company can enhance teamwork and improve task execution.

Ultimately, this project aims to empower MWW to deliver not only high-quality but best quality services efficiently to its customers. By optimizing employee assignments and leveraging the expected learning trajectories of individuals and teams, the company can enhance service delivery quality, customer satisfaction, and overall operational efficiency. In the following sections, we will detail the methodologies employed, the analysis conducted, and the insights gained throughout this project. By undertaking this project, we seek to support MWW in their pursuit of excellence and continuous improvement, enabling them to deliver exceptional services and drive customer satisfaction to new heights.

2. ER Diagram



3. Task-1

Task: Start by visualizing the learning curves of individual employees. It is likely that the personnel learn overtime and thus need less time for repeating tasks.

The calculation of the learning over time was done with the help of MS SAP DWC datasets *MwwMsAssignedEmployees* and *MwwMsCompletedTasks*. A join of these two datasets facilitated the calculations.

Tool Used: BigQuery, Looker Studio

Analysis:

To obtain individual learning over time, we have joined the two tables ***MwwMsAssignedEmployees*** and ***MwwMsCompletedTasks*** with join condition on ***'ServiceTaskId'*** *(Common column in both tables)*. After getting all the employees and ***'ServiceTaskId'*** from the first table, we have then calculated the time spent criteria to obtain individual learning over time in a separate column called ***TimeOnEachTask***. To achieve this, we are dividing *'**TotalWorkingHours**'* by *'**Quantity**'* which will then calculate the individual efforts for each employee, for each task. Query is attached below.

Assumption: Each person on a team has equal contribution regarding the amount of time.

Note: **TimeOnEachTask** column has been rounded to two places decimal to make it more human readable.

3.1 Query used

```
select
AE.ServiceEmployeeId,
AE.ServiceTaskId,
CT.Date,
CT.TaskType,
CT.Quantity,
CT.TotalWorkingHours,
ROUND(CT.TotalWorkingHours / CT.Quantity, 2) as TimeOnEachTask
from (
MwwMsSAPData.MwwMsAssignedEmployees as AE
join MwwMsSAPData.MwwMsCompletedTasks as CT
on CT.ServiceTaskId = AE.ServiceTaskId
);
```

The figure shows a preview of the test dataset formed with the help of the above query, the **'TimeOnEachTask'** attribute specifying the time taken by each employee for a particular task. This encompasses all information required for learning curve visualizations for individual employees.

Query Output:

The below query output is very single employee (SE100329) and worked on single task (Bike repair), which clearly represents the learning over time by gradually reducing the amount of time spent over time.

Row	ServiceEmployeeId	ServiceTaskId	Date	TaskType	Quantity	TotalWorkingHours	TimeOnEachTask
19	SE100329	SO100084	2001-05-29	Bike repair	11	5.35	0.49
20	SE100329	SO100090	2001-06-02	Bike repair	28	13.6	0.49
21	SE100329	SO100100	2001-06-18	Bike repair	19	9.21	0.48
22	SE100329	SO100116	2001-07-23	Bike repair	22	10.64	0.48
23	SE100329	SO100140	2001-08-25	Bike repair	24	11.6	0.48
24	SE100329	SO100170	2001-09-17	Bike repair	26	12.54	0.48
25	SE100329	SO100197	2001-10-12	Bike repair	35	16.85	0.48
26	SE100329	SO100248	2001-12-04	Bike repair	19	9.13	0.48
27	SE100329	SO100268	2001-12-20	Bike repair	20	9.59	0.48
28	SE100329	SO100272	2001-12-22	Bike repair	10	4.78	0.48
29	SE100329	SO100290	2002-01-07	Bike repair	10	4.77	0.48
30	SE100329	SO100305	2002-01-21	Bike repair	11	5.24	0.48
31	SE100329	SO100318	2002-01-28	Bike repair	9	4.28	0.48
32	SE100329	SO100339	2002-03-03	Bike repair	23	10.92	0.47
33	SE100329	SO100387	2002-05-08	Bike repair	19	9.01	0.47

Results per page: 200 1 – 119 of 119

Figure 1 - BigQuery Output representing individual learning over time.

3.2 Visualization Using Looker Studio

As we have seen previously, learning of employee using BigQuery. Now, using Looker Studio we have created visualized representation of the same data in the form of Charts.

Visualization reports were prepared, showcasing the learning over time for individual employees and for all employees, for each task.

We have used Google Looker Studio tool to prepare these charts. Where:-

X-axis: Date and EmployeeServiceID

Y-axis: TimeOnEachTask (Timespent by each employee on a particular task)

<u>Individual Learning graphs – single employee, single task:</u>

Below is the learning curve of employee (SE100329) for a single task (Bike repair) which clearly represents the declning graph representing reduced time spent over time.

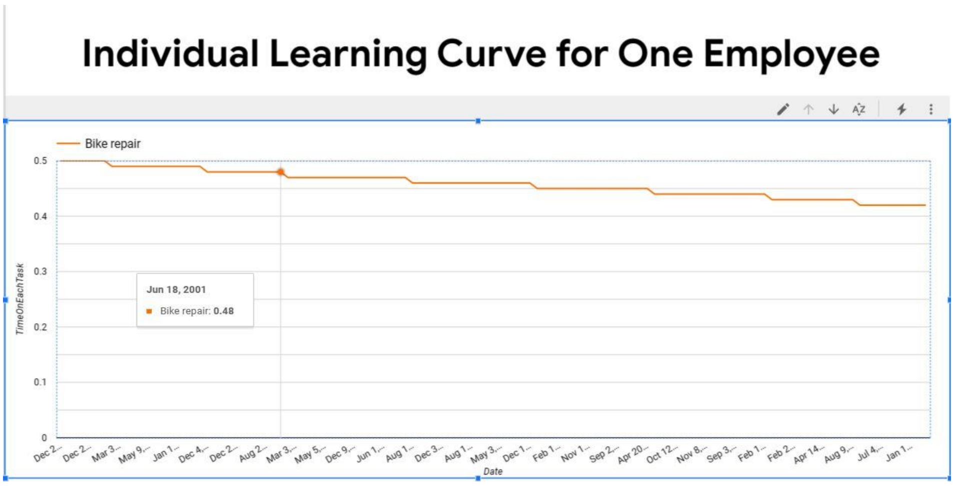


Figure 2 - Learning Curve of Employee: SE100329 for Task: Bike repair on June 18, 2001

<u>Individual employee learning over time:</u>

Below chart shows learning curve of ServiceEmployeeID: SE100329 who spent 3.15hours on “Charging station check”, 3.89hours on “Electronics repair”, 1.79hours on “Vehicle maintenance” and 0.48hours on “Bike repair” on Oct 12, 2001.

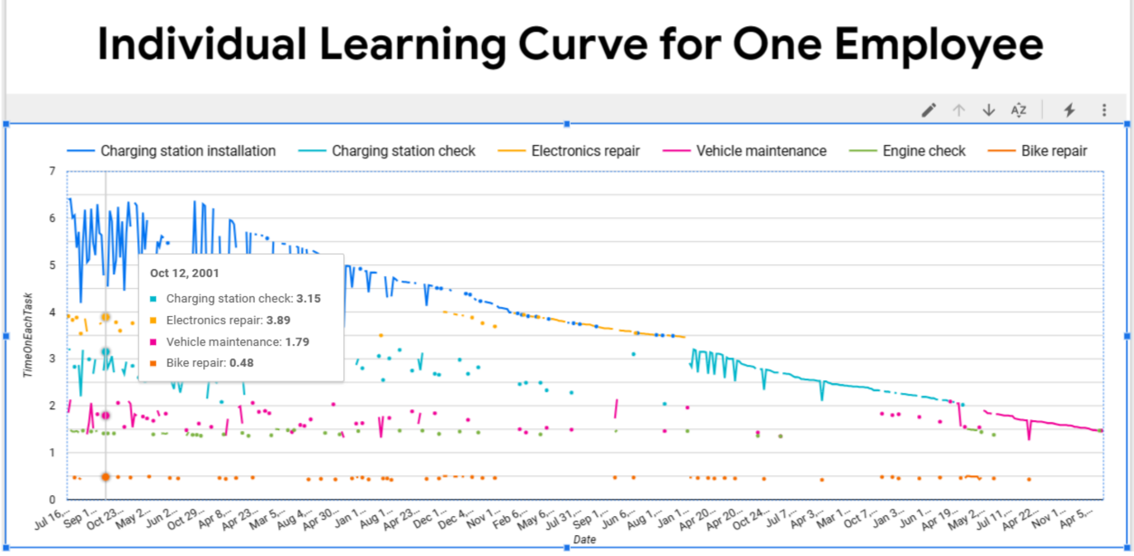


Figure 3 - Learning curve of ServiceEmployeeID: SE100329 on Oct 12, 2001

On the other hand, the second chart reprentsshat time has been reduced and there is a learning achieved over time. Now the same employee spent less time for the same tasks such as 1.26hours on “Vehicle maintenance” and 0.43hours on “Bike repair” on May 1, 2016.

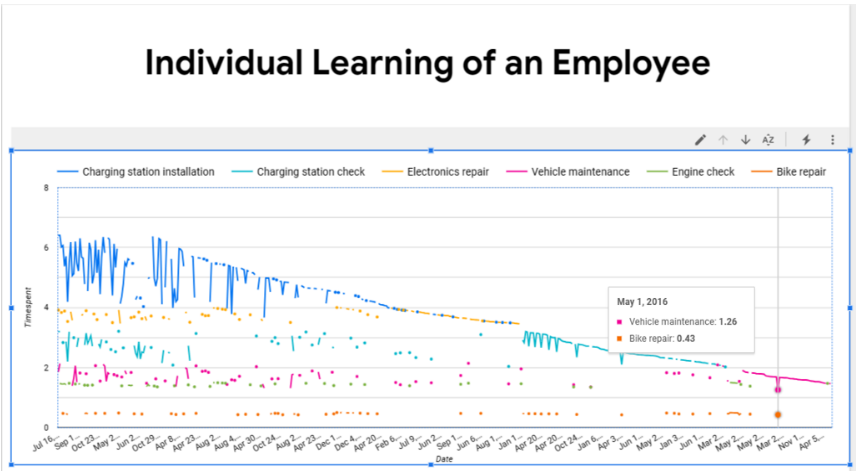


Figure 4 - Learning curve of ServiceEmployeeID: SE100329 on May 1, 2016.

Learning Curve of all employees over time:

The below chart graph will talk about the cumulative learning for each task of all employees on Dec 5, 2004 that they spent 30.89hours on “Charging station installation”,10.97hours “Charging station check”, 6.42hours on “Vehicle maintenance”, 7.33hours on “Electronics repair”, 5.34hours on “Engine check” and 2.46hours on “Bike repair”.

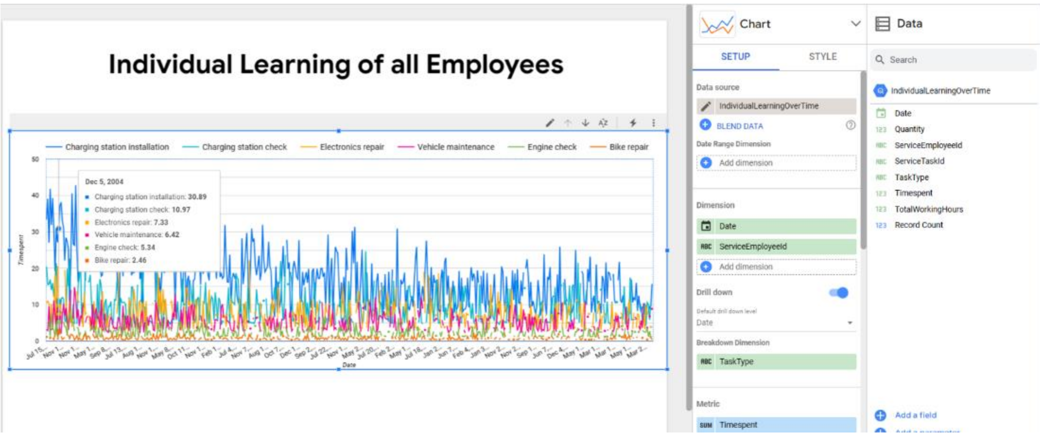


Figure 5 - Learning curve of all service employee in Dec 5, 2004

Where the second graph, mentioned below, reflects lower values for Sep 10, 2006, we will observe that all the values has been reduced over time and thereby it is clearly visible that chart represents the learning over time.

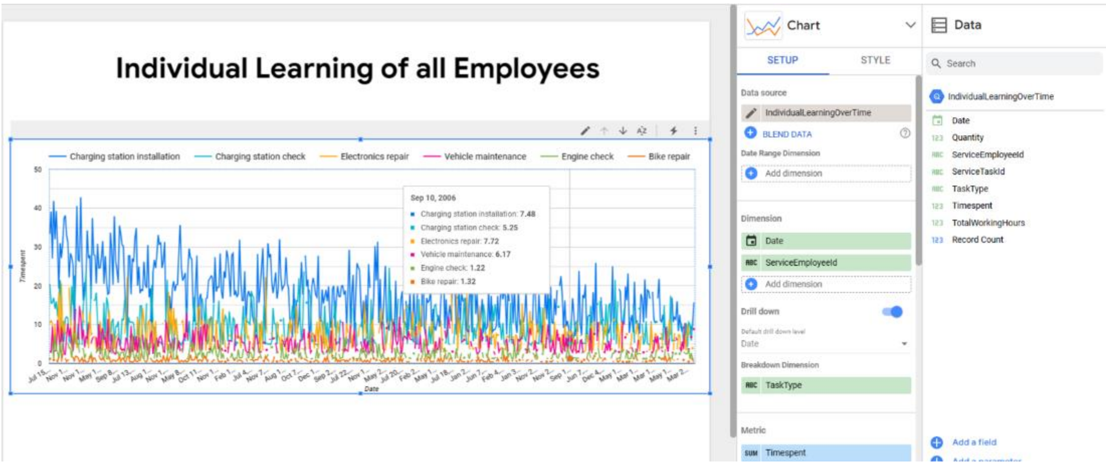


Figure 6 - Learning curve of all service employee in Sep 10, 2006

Link to the Looker Studio Report:

https://lookerstudio.google.com/s/prVMMF3jGAo

4. Task-2

<u>Task:</u> Some jobs require more than one employee. Evaluate whether there are teams whose work is more effective than the work of other teams. The distance to the client site might also play a role in this regard.

4.1 Task scrutiny

From the given data we have extracted some information which is:

Service Point: All service points of the company. Multiple employees are assigned at different Service Points. Our 25 Employees belong to a single Service Point.

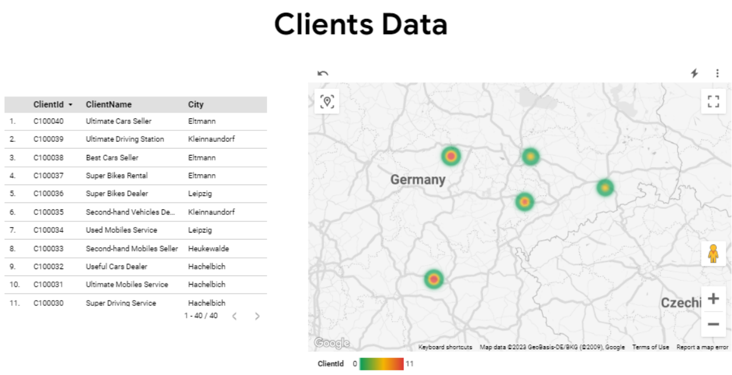
Service Employees: Information of all available employees and their information like Name, gender and working experience. All employees belong to only one Service Point Id.

Open Contracts: A contract can span over multiple years, for instance, all the contracts span over almost 3 years. A contract belongs to a single client. A client can have multiple contracts. Each contract can consist of multiple tasks.

Completed Tasks: Information on all the completed tasks. The task type and quantity of tasks performed and total Working Hours spent on completing the task. We can calculate the average time to complete the task. We can find the assigned employee for a particular task from the Assigned Employees relationship. We have also found that some tasks are performed individually and some are performed in a team which will help us to evaluate team performance.

To Evaluate the Team performances and see the effect of Distance to client site over Employee performance we have formulated some Relations and ran some SQL queries using BigQuery Tool. The results are then visualized using Looker Studio Tool.

Client Table Update: Client Relation is updated with client location like City and Postal Code. All our Service Employees belong to a single Service Point which is Königsee (7426). All clients belong to 5 cities. We have used Google Distance Matrix API to calculate travel distances and time needed by car between the origin (e.g., the service point) and the client's location which is the destination of Service Employees. We have added this information to our client relationships and named it *'DistancesAndTimeServiceToClient'*. The locations of clients are shown below as Heat Map.



	ClientId	ClientName	City
1.	C100040	Ultimate Cars Seller	Eltmann
2.	C100039	Ultimate Driving Station	Kleinnaundorf
3.	C100038	Best Cars Seller	Eltmann
4.	C100037	Super Bikes Rental	Eltmann
5.	C100036	Super Bikes Dealer	Leipzig
6.	C100035	Second-hand Vehicles De...	Kleinnaundorf
7.	C100034	Used Mobiles Service	Leipzig
8.	C100033	Second-hand Mobiles Seller	Heukewalde
9.	C100032	Useful Cars Dealer	Hachelbich
10.	C100031	Ultimate Mobiles Service	Hachelbich
11.	C100030	Super Driving Service	Hachelbich

Figure 7 - Client Locations

Completed Task with Travelling Info: We have added the traveling info to the completed tasks to know how much distance the service employee has traveled to the client site to complete that task. Later, we will examine the effect of distance on Service Employees' performance.

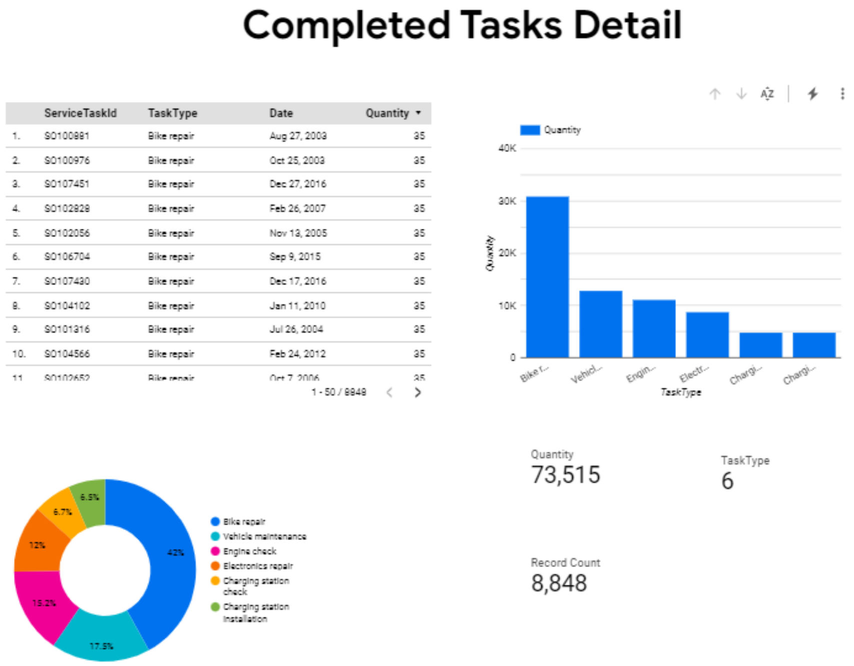


Figure 8 - Completed task distribution

4.2 Dataset and SQL queries

<u>Tools Used:</u> BigQuery, Google Distance Matrix API, Looker Studio

To know if there are tasks that are done by more than one employee, we have run the following query. By adjusting the value to 3 in the last step of the query, no outcome was achieved. However, by setting it to 2, we obtained results, indicating that a maximum of two employees have worked within a team.

<u>Query used:</u>

```sql
SELECT ServiceTaskId
FROM `proj-iv-maintenance-service.MwwMsSAPData.MwwMsAssignedEmployees`
GROUP BY ServiceTaskId
HAVING COUNT(DISTINCT ServiceEmployeeId) > 1;
```

Such tasks in which More than one employee has worked are:

- Charging Station Installation
- Charging Station Check
- Vehicle Maintenance

Completed tasks with Travelling Info:

We have joined *CompletedTask* Relation with *DistanceAndTimeServicePointToClient* Relation. Another column called *'AverageTimeToCompleteTheTask'* is added which is the ratio of Total Working hours to complete the task and Quantity. *'DistanceTakenToReachTheSite'* and *'AverageTimeToReachTheSite'* are included in the final relation.

<u>Query used:</u>

```
SELECT
  CT.*,
  DTPC.City AS ClientCity,
  DTPC.PostalCode,
  DT.Distance__Km_ AS DistanceTakenToReachSite,
  DT.Time__hrs_ AS TimeTakenToReachSite,
FROM
  `proj-iv-maintenance-service.MwwMsSAPData.MwwMsCompletedTasks` AS CT
JOIN
  `proj-iv-maintenance-service.MwwMsSAPData.MwwMsOpenContracts` AS OC
ON
  CT.ContractId = OC.ContractId
JOIN
  `proj-iv-maintenance-
service.SampleTestingDataset.DistancesAndTimeServicePointToClient` AS DT
ON
  OC.ClientId = DT.ClientId
JOIN
  `proj-iv-maintenance-
service.SampleTestingDataset.DistancesAndTimeServicePointToClient` AS DTPC
ON
  OC.ClientId = DTPC.ClientId
ORDER BY
  ServiceTaskId ASC;
```

Given below is the preview of dataset *CompletedTaskWithTravellingInfo* which contains *'ServiceTaskId'*, *'TaskType'*, *'ClientCity'*, *'PostalCode'*, *'DistanceTakenToReachSite'*

Row	ServiceTaskId	ContractId	Date	TaskType	Quantity	TotalWorkingHours	ClientCity	PostalCode	DistanceTakenToReachSi	TimeTakenToReachSite
1	SO100001	CT100001	2000-12-09	Bike repair	17	8.5	Leipzig	4157	193	2.1
2	SO100002	CT100002	2000-12-14	Electronics repair	3	12.0	Eltmann	97483	134	1.7
3	SO100003	CT100002	2000-12-14	Engine check	3	4.5	Eltmann	97483	134	1.7
4	SO100004	CT100002	2000-12-14	Bike repair	20	10.0	Eltmann	97483	134	1.7
5	SO100005	CT100002	2000-12-14	Bike repair	11	5.49	Eltmann	97483	134	1.7
6	SO100006	CT100002	2000-12-16	Bike repair	9	4.48	Eltmann	97483	134	1.7
7	SO100007	CT100001	2000-12-16	Bike repair	35	17.45	Leipzig	4157	193	2.1
8	SO100008	CT100002	2000-12-16	Electronics repair	2	7.99	Eltmann	97483	134	1.7
9	SO100009	CT100001	2000-12-19	Engine check	7	10.5	Leipzig	4157	193	2.1
10	SO100010	CT100002	2000-12-19	Bike repair	10	4.97	Eltmann	97483	134	1.7
11	SO100011	CT100001	2000-12-22	Bike repair	21	10.43	Leipzig	4157	193	2.1

Results per page: 50 1 – 50 of 8848

Table 1 - Dataset - CompletedTaskWithTravellingInfo

Employee Info is added to Completed Tasks with Travel Info by joining *'CompletedTaskWithTravellingInfo'* and ServiceEmployee and AssignedEmployee. We have added the Employee Name and Id. We have excluded all the individual tasks and consider only those tasks which require more than one employee. *te1.ServiceEmployeeId < te2.ServiceEmployeeId* condition is used to remove redundant entries.

Query used:

```
SELECT
  ct.ServiceTaskId,
  te1.ServiceEmployeeId AS Employee1Id,
  te1.Name AS Employee1Name,
  te2.ServiceEmployeeId AS Employee2Id,
  te2.Name AS Employee2Name,
  ct.TaskType,
  ct.Quantity,
  ct.TotalWorkingHours,
  ROUND((ct.TotalWorkingHours / ct.Quantity), 2) AS AverageTime,
  ct.ClientCity,
  ct.DistanceTakenToReachSite,
  ct.TimeTakenToReachSite,
FROM
`proj-iv-maintenance-
service.SampleTestingDataset.CompletedTaskWithTravellingInfo` AS ct
JOIN
  `proj-iv-maintenance-service.MwwMsSAPData.MwwMsAssignedEmployees` AS ae1 ON
ct.ServiceTaskId = ae1.ServiceTaskId
JOIN
  `proj-iv-maintenance-service.MwwMsSAPData.MwwMsAssignedEmployees` AS ae2 ON
ct.ServiceTaskId = ae2.ServiceTaskId
JOIN
  `proj-iv-maintenance-service.MwwMsSAPData.MwwMsServiceEmployees` AS te1 ON
ae1.ServiceEmployeeId = te1.ServiceEmployeeId
JOIN
  `proj-iv-maintenance-service.MwwMsSAPData.MwwMsServiceEmployees` AS te2 ON
ae2.ServiceEmployeeId = te2.ServiceEmployeeId
WHERE
  te1.ServiceEmployeeId < te2.ServiceEmployeeId
```

We then created three test datasets, namely *ChargingStationCheckWithTravellingInfo*, *ChargingStationInsatallationWithTravellingInfo* and *VehivleMaintenanceWithTravellingInfo* for each of the three multi team tasks that we identified earlier.

Following Query is used to filter out the Tasks from the *MultiEmployeeTask* relation

```
SELECT
  *
FROM
  `proj-iv-maintenance-service.SampleTestingDataset.MultiEmployeeTasks`
  where TaskType='Charging station check'
```

For our analysis we segregated teams based on two criteria, task type and gender of employees.

Tasks specific teams:

- Team - ChargingStationInstallation
- Team - ChargingStationCheck
- Team - VehicleMaintenance

During our analysis we noticed that members of teams have worked together multiple times. We calculated the number of times two team members worked together, then we calculated their Average time and took average of distance travelled by the team. Given below is the query used to segregate the team *ChargingStationInstallation.*

Query Used:

```
SELECT
  CONCAT(Employee1Name, ' and ', Employee2Name) AS EmployeesName,
  CONCAT(Employee1Id, ' and ', Employee2Id) AS EmployeesId,
  COUNT(*) AS Quantity,
  ROUND(AVG(AverageTime), 2) AS TaskAverageTime,
  ROUND(AVG(DistanceTakenToReachSite), 2) AS AvgDistanceTakenToReachSite,
  ROUND(AVG(TimeTakenToReachSite), 2) AS AvgTimeTakenToReachSite
FROM
  `SampleTestingDataset.ChargingStationInstallationTaskInfo`
GROUP BY
  EmployeesName,
  EmployeesId
ORDER BY
  TaskAverageTime ASC;
```

Same is done for Charging Station Check and Vehicle Maintenance.

Final Team Evaluation dataset:

Finally, all teams of Task are joined and assigned with unique TeamId. All teams are evaluated based on their TaskAverageTime. Following query is used to join Tables *TeamChargingStationInstallation*, *TeamChargingStationCheck*, *TeamVehicleMaintenance*.

```
SELECT
  CONCAT('TIDCSC',ROW_NUMBER() OVER (ORDER BY EmployeesId)) AS TeamId,
  'Charging station check' AS TaskType,
  EmployeesName,
  EmployeesId,
  Quantity,
  TaskAverageTime,
  AvgDistanceTakenToReachSite,
  AvgTimeTakenToReachSite
FROM
  SampleTestingDataset.TeamChargingStationCheck
UNION ALL
SELECT
  CONCAT('TIDCSI',ROW_NUMBER() OVER (ORDER BY EmployeesId)) AS UniqueId,
  'Charging station installation' AS TaskType,
  EmployeesName,
  EmployeesId,
  Quantity,
  TaskAverageTime,
  AvgDistanceTakenToReachSite,
  AvgTimeTakenToReachSite
FROM
  SampleTestingDataset.TeamChargingStationInstallation
UNION ALL
SELECT
  CONCAT('TIDVM',ROW_NUMBER() OVER (ORDER BY EmployeesId)) AS UniqueId,
```

```
  'Vehicle maintenance' AS TaskType,
  EmployeesName,
  EmployeesId,
  Quantity,
  TaskAverageTime,
  AvgDistanceTakenToReachSite,
  AvgTimeTakenToReachSite
FROM
  SampleTestingDataset.TeamVehicleMaintenance;
```

JOB INFORMATION RESULTS JSON EXECUTION DETAILS EXECUTION GRAPH

Row	TeamId	TaskType	EmployeesName	EmployeesId	Quantity	TaskAverageTime	AvgDistanceTakenTo	AvgTimeTakenToReachSit
1	TIDVM89	Vehicle maintenance	Craig Stevenson and Jack Stev...	SE100332 and SE100352	2	1.11	201.0	2.24
2	TIDVM67	Vehicle maintenance	Craig Brown and Michelle Myers	SE100331 and SE100351	1	1.13	113.0	1.55
3	TIDVM82	Vehicle maintenance	Craig Stevenson and Martha M...	SE100332 and SE100345	7	1.13	210.86	2.27
4	TIDVM68	Vehicle maintenance	Craig Brown and Jack Stevenson	SE100331 and SE100352	2	1.15	128.0	1.6
5	TIDVM51	Vehicle maintenance	Craig Brown and Michael O'Neil	SE100331 and SE100335	7	1.15	156.86	1.86
6	TIDVM87	Vehicle maintenance	Craig Stevenson and John Jac...	SE100332 and SE100350	1	1.16	193.0	2.1
7	TIDVM66	Vehicle maintenance	Craig Brown and John Jackson	SE100331 and SE100350	3	1.16	223.33	2.42
8	TIDVM141	Vehicle maintenance	Michael O'Neil and Tina Myers	SE100335 and SE100349	3	1.17	130.0	1.64
9	TIDVM120	Vehicle maintenance	Antoni Jackson and Martha Mc...	SE100334 and SE100345	2	1.18	123.5	1.63

Results per page: 50 1 – 50 of 863

Table 2: Team Evaluation Dataset

4.3 Team Performance Evaluation

We have generated data of all teams which performed Charging Station Installation, Charging Station check and Vehicle Maintenance. Teams are arranged based on Average time in ascending order. Teams with less average time to complete the tasks are considered as more efficient.

X-Axis = Team Members

Y-Axis = Average Time to Complete the task.

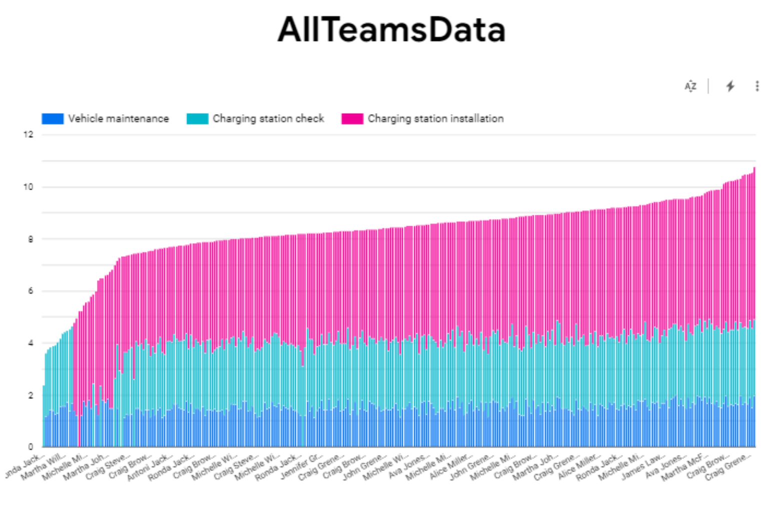


Figure 9: Team Efficiency - Average Time Visualization

Some Team members have performed all three tasks, some have performed only two tasks and some have performed only one task. Craig Stevenson and Jack Myers have performed Charging Station Installation with an average time of 3.4 hours, Charging Station Check with an average time of 2.57 hours, Vehicle Maintenance with an average Time of 1.48 hours.

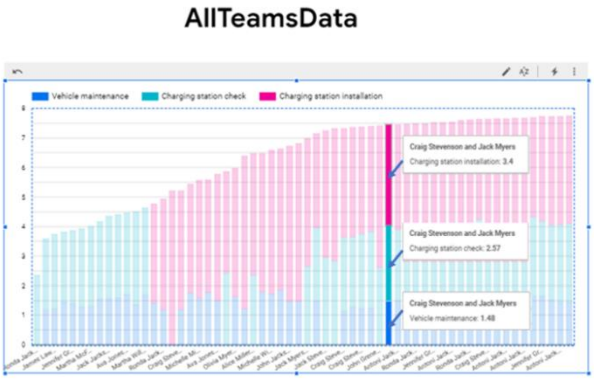


Figure 10: Average Time - Craig Stevenson & Jack Myers

Some Teams have performed two tasks for example Ava Jones and Jack Stevenson have done Charging Station Installation with Average time to complete a task of 4.3 hours and Vehicle Maintenance with an Average time to complete task of 1.9 hours.

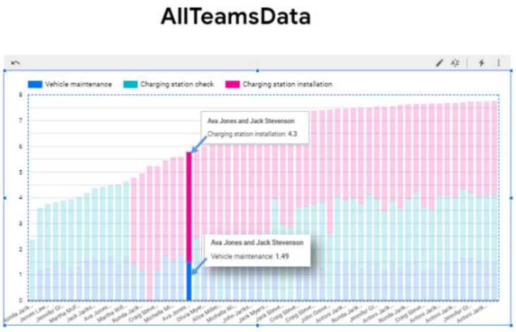


Figure 12 - Avg Time- Ava Jones & Jack Stevenson

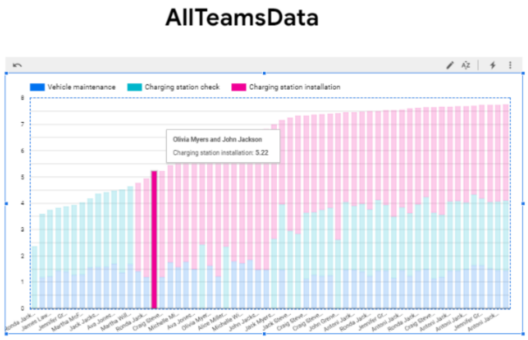


Figure 11 - Avg Time Olivia Myers and John Jackson

Some Teams has performed only one task for example Olivia Myers and John Jackson has performed only Charging Station Installation with average time of 5.22 hours to complete the task. We have used BigQueryML to predict the Average time to complete the task for those teams which have not performed that task.

Effect of Distances on Team Efficiency:

We have seen the effect of distance on teams by filtering the task type to a single task type for instance Charging station Installation and setting the quantity to a particular number for instance 5 to see the effects of distance travelled by the team employees to the client site, on the team performance. For example, Michelle Wiliams and James Lawson has travelled average distance of 145.2km and their performance metric, average time to complete the task is 2.4 hours. While Ava Jones and Martha McFinnegan has travelled 168.4km and their performance metric, average time to complete the task is 2.87 hours. So, it shows that distance affects their performance but for some team's distance does not affect their performance. For instance, Craig Brown and

Michael Jackson have travelled average distance of 174.4km and yet their average time to complete the task is 2.51 hour.

It can be possible that those teams for which distance does not affect the performance can be of opposite Gender and they like to work together.

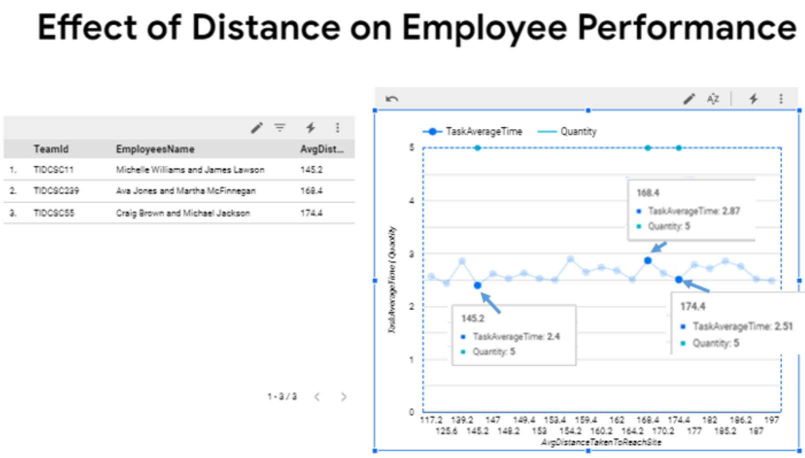


Figure 13: Client Site Distance Analysis Graph

Gender specific teams:

- ChargingStationCheck-GenderInfo
- ChargingStationInstallation-GenderInfo
- VehicleMaintenance-GenderInfo

Given below is the query used to segregate the team ChargingStationInstallation with GenderInfo. Teams with only male employees were labelled as ‘Male Team’, teams with only female employees were labelled as ‘Female Team’ and the teams with both male and female employees were labelled as ‘Couple Team’. Here also the time taken to complete the task, distance to reach client site and time taken to reach client site were calculated as the average of the corresponding values for both employees in the team and the total working hours were computed as the sum of the total working hours of both the employees.

This segregation was done to analyze the effect of team gender on team performance and to identify the best teams.

Query used:

```
SELECT
    CONCAT(Employee1Name, ' and ', Employee2Name) AS Team,
    CASE
        WHEN e1.Gender = 'M' AND e2.Gender = 'M' THEN 'Male Team'
        WHEN e1.Gender = 'F' AND e2.Gender = 'F' THEN 'Female Team'
        ELSE 'Couple Team'
    END AS TeamGender,
    COUNT(*) AS Quantity,    ROUND(SUM(TotalWorkingHours), 2) AS
TotalWorkingHours,
    ROUND(AVG(AverageTime), 2) AS AverageTime,
    ROUND(AVG(DistanceTakenToReachSite), 2) AS AvgDistanceTakenToReachSite,
    ROUND(AVG(TimeTakenToReachSite), 2) AS AvgTimeTakenToReachSite
FROM
    SampleTestingDataset.ChargingStationInstallationWithTravellingInfo
```

```
JOIN
     `MwwMsSAPData.MwwMsServiceEmployees` AS e1 ON Employee1Id =
e1.ServiceEmployeeId
JOIN
     `MwwMsSAPData.MwwMsServiceEmployees` AS e2 ON Employee2Id =
e2.ServiceEmployeeId
GROUP BY
     Team, TeamGender
ORDER BY
     Quantity DESC;
```

<u>Query output:</u>

Row	Team	TeamGender	Quantity	TotalWorkingHours	AverageTime	AvgDistanceTakenTo	AvgTimeTakenToRea
1	Michelle Miller and Craig Brown	Couple Team	24	449.51	5.94	145.42	1.78
2	Craig Brown and Craig Stevens...	Male Team	19	373.92	4.87	161.42	1.9
3	Michelle Williams and Craig St...	Couple Team	18	308.12	4.86	143.22	1.75
4	Michelle Miller and Alice Miller	Female Team	16	299.67	5.53	174.19	2.0
5	Michelle Williams and Michelle ...	Female Team	16	340.75	5.9	152.13	1.83
6	Ronda Jackson and John Gren...	Couple Team	15	215.04	4.76	155.07	1.85
7	Ronda Jackson and Craig Gren...	Couple Team	14	206.29	4.87	145.21	1.76
8	Alice Miller and Michael O'Neil	Couple Team	14	199.97	5.75	194.29	2.15
9	Craig Brown and Jennifer Gren...	Couple Team	14	207.45	4.33	149.0	1.82
10	Michelle Williams and Craig Br...	Couple Team	13	329.72	5.86	138.23	1.73
11	Michelle Miller and Jennifer Gr...	Female Team	13	184.73	4.33	143.77	1.76
12	Michelle Williams and Alice Mil...	Female Team	12	180.52	5.32	176.83	1.99
13	Michael O'Neil and Ronda Jack...	Couple Team	12	124.89	4.96	169.83	1.95

Figure 14 - Charging Station Installation team with Gender Info

So for each task type we have seen the participation of Team of employees based on their Gender and we have seen that most of the employee has worked as a couple in a team. So most of the employee like to work with opposite genders employee.

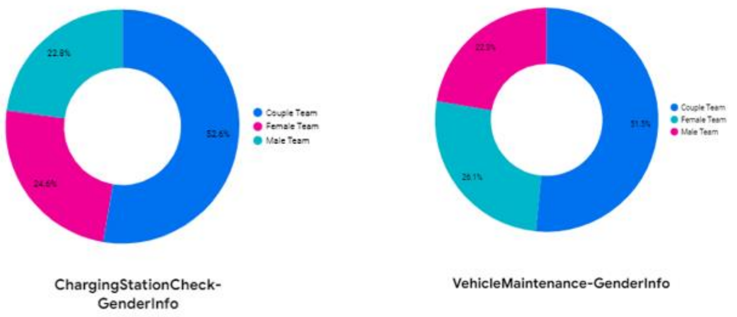


Figure 15 - Task distribution visualization based on gender info

5. Task-3

Task: Develop a simple simulation to create new tasks and assign them to the assumingly best workers or teams. Demonstrate the board the benefits that a data-driven assignment of employees has on the overall efficiency. This task was divided into two parts:

- Creation of a machine learning model which considers all the individual employees/teams and finds out the time taken by them to complete each task.
- Development of a simulation model (flask application) that generates a random task from the given list of tasks using a button and then returns the list of employees/teams and their time for that specific task arranged in ascending order of the time taken to complete the task.

Tools used in GCP: BigQuery, App Engine

Tech. stack used: Python (Flask), HTML, CSS (Bootstrap)

5.1 ML model

We have broken down this task into two stages. Initially, we developed a machine learning model that takes into account each individual employee/team and determines the time required by them to finish each task. Subsequently, we built a simulation model in the form of a Flask application. This application allows users to randomly select a task from a provided list by clicking a button. The application then generates and presents a list of employees/teams along with their corresponding completion times for the selected task. The list is organized in ascending order based on the time taken to complete the task.

We utilized BigQuery ML to construct the machine learning model, which was seamlessly integrated with our flask application. The entire application was then deployed on Google Cloud's App Engine, effectively connecting the power of BigQuery with the user-friendly interface of our web application.

5.1.1 Model version 1

5.1.1.1 Dataset for ML model

Simplification: Distance to the client site is irrelevant for our machine learning model because for the same client location, the distance for each employee/team is same. Hence this value was not considered for preparing the data for the ML model. Feature

- **TeamID:** This represents the unique identifier for each employee/team involved in the task assignments.
- **TaskType:** This refers to the type of task being assigned to the teams.
- **UniqueTeams:** This refers to the teams for completed task assignments.
- **AverageTimeToCompleteTheTask:** This represents the average time taken by the teams to complete the assigned task.

Query used:

```
SELECT
  CONCAT('T', LPAD(CAST(ROW_NUMBER() OVER (ORDER BY c.TaskType,
c.NameOfServiceEmployee) AS STRING), 4, '0')) AS TeamId,
  c.TaskType,
  c.NameOfServiceEmployee AS UniqueTeams,
  AVG(c.AverageTimeToCompleteTheTask) AS AverageTimeToCompleteTheTask
FROM (
  SELECT
    TaskType,
    NameOfServiceEmployee,
    DistanceTakenToReachSite,
    TimeTakenToReachSite,
    AverageTimeToCompleteTheTask
  FROM MwwMsSAPData.CompletedTasksWithEmployeeTravellingInfo
  GROUP BY TaskType, NameOfServiceEmployee, DistanceTakenToReachSite,
TimeTakenToReachSite, AverageTimeToCompleteTheTask
) c
GROUP BY c.TaskType, c.NameOfServiceEmployee
```

Query output:

SCHEMA DETAILS PREVIEW LINEAGE

Row	TeamId	TaskType	UniqueTeams	AverageTimeToCompleteTheTask
1	T0013	Bike repair	John Jackson	0.41179421482842371
2	T0009	Bike repair	Jack Stevenson	0.49440466742914768
3	T0002	Bike repair	Antoni Jackson	0.40443936777503675
4	T0014	Bike repair	Martha Johnson	0.48057776151624088
5	T0001	Bike repair	Alice Miller	0.39988770203904062
6	T0022	Bike repair	Olivia Myers	0.4886807069273697
7	T0003	Bike repair	Ava Jones	0.41183641130581056
8	T0021	Bike repair	Michelle Williams	0.45868655506355221
9	T0024	Bike repair	Tina Myers	0.49134569177111403
10	T0017	Bike repair	Michael Jackson	0.48032649679742973
11	T0015	Bike repair	Martha McFinnegan	0.4198547866782481
12	T0011	Bike repair	Jennifer Grenelli	0.46786349396193289
13	T0005	Bike repair	Craig Grenelli	0.40152790419113055
14	T0010	Bike repair	James Lawson	0.48178063096875923

Figure 16: Model Data Generation Query Output

Split Data into Train and Test

To ensure proper model training and evaluation, we saved the obtained result into a table named **"MwwMsSAPData.Inputdata"**. This table will serve as our dataset for training the machine learning model.

In addition, we performed a data split to create separate **train** and **test** datasets. This allows us to assess the performance of the model on unseen data and evaluate any potential loss. The following query demonstrates the data split:

Query used:

```
-- Split the dataset into a training set and a test set
CREATE OR REPLACE TABLE MwwMsSAPData.Inputdata_train AS
SELECT TaskType,
UniqueTeams,
AverageTimeToCompleteTheTask
FROM MwwMsSAPData.Inputdata
WHERE MOD(ABS(FARM_FINGERPRINT(TeamId)), 10) < 8; -- 80% for training

CREATE OR REPLACE TABLE MwwMsSAPData.Inputdata_test AS
SELECT
TaskType,
UniqueTeams,
AverageTimeToCompleteTheTask
FROM MwwMsSAPData.Inputdata
WHERE MOD(ABS(FARM_FINGERPRINT(TeamId)), 10) >= 8; -- 20% for testing
```

5.1.1.2 ML model

Model Selection

For our machine learning model, we selected a classic **linear regression** approach. This choice was based on the nature of our data, which includes three categorical features (TeamID, UniqueTeams, and TaskType) and a continuous target variable (AverageTimeToCompleteTheTask).

Linear regression is a suitable choice for this scenario as it allows us to model the relationship between the input features and the target variable by estimating the coefficients that define a linear equation. In our case, we aim to predict the average time taken to complete a task based on the categorical features of the team assigned to the task.

By using linear regression, we can capture the linear relationships and patterns in the data, allowing us to make predictions and understand the impact of each categorical feature on the target variable. This approach provides interpretability and insights into how different teams and task types affect the average time to complete a task.

Model Creation and Training

The ML model was created using BigQueryML. The creation and training were done as below:

Query used:

```
-- Create and train the regression model
CREATE OR REPLACE MODEL MwwMsSAPData.TimePredictionModel
OPTIONS(model_type='linear_reg') AS
```

```
SELECT
  TaskType,
  UniqueTeams,
  AverageTimeToCompleteTheTask AS label
FROM MwwMsSAPData.Inputdata_train;
```

Model Evaluation

The **training loss** is a metric that quantifies the discrepancy between the predicted values and the actual values of the target variable during the training phase of a machine learning model. It measures how well the model is able to fit the training data.

The **evaluation loss**, also known as the validation loss, is a metric that measures the performance of the trained model on a separate dataset called the evaluation dataset or validation dataset. This dataset consists of data that the model hasn't seen during training and serves as a proxy for the model's ability to generalize to new, unseen data.

In the below figure, the orange line shows the training loss and the purple line shows the evaluation loss. We see that the training and evaluation loss come up to 0.158 and 1.578. Also, the learning rate is 0.4 in the final iteration. After some analysis we found out that we can improve the evaluation loss.

In the provided **figure** (figure no.), a line graph is displayed with two lines: an **orange line** representing the training loss and a **purple line** representing the evaluation loss. The values on the y-axis represent the loss values, while the x-axis represents the iterations or epochs of the training process.

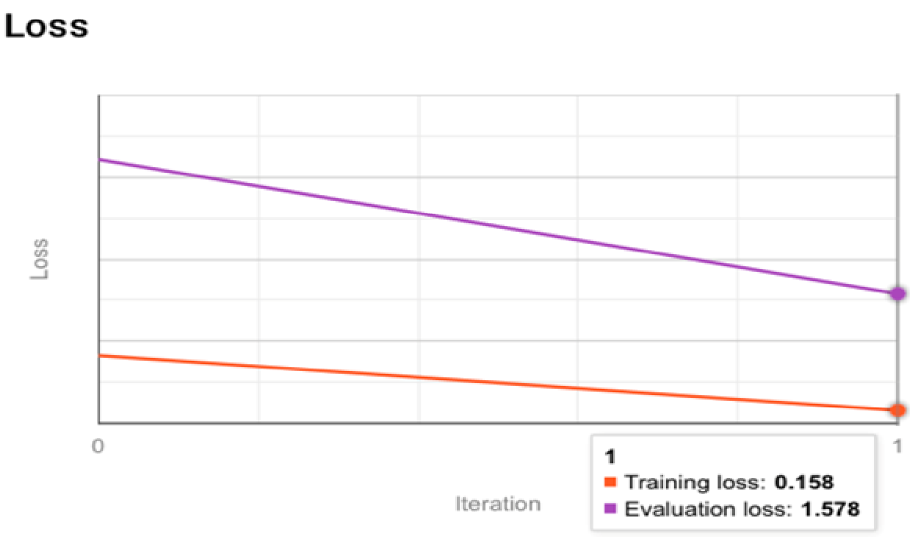


Figure 17: Training & Evaluation Loss of ML Model Version 1

The training loss is indicated by the orange line and has a value of 0.158. This value reflects the discrepancy between the predicted and actual values of the target variable during the training phase.

The evaluation loss is represented by the purple line and has a value of 1.578. This loss value is calculated on a separate evaluation dataset, which contains data that the model hasn't seen during training. The evaluation loss provides an estimate of how well the model is expected to perform on unseen data. In this case, the evaluation loss indicates that the model may not generalize as effectively to new data as it does to the training data.

Based on the analysis conducted, it has been determined that there is room for improvement in the evaluation loss. This implies that the model may not be generalizing well to new, unseen data. Further steps can be taken to refine the model, such as adjusting hyperparameters, increasing the complexity of the model, or gathering additional relevant data. By iteratively evaluating and improving the model, the aim is to reduce the evaluation loss and enhance the model's predictive performance on unseen data.

5.1.2 Model version 2 - Final Model

5.1.2.1 Dataset for ML model

Improvement: To improve the model, a new dataset was prepared by refining the feature space. One enhancement involved removing the TeamID feature from the dataset, potentially leading to a more effective model.

Additionally, the dataset was once again split into training and testing subsets. This division allows for the evaluation of the model's performance on unseen data and helps assess its generalization capabilities.

By refining the feature space and creating new training and testing datasets, the aim is to enhance the model's ability to capture meaningful patterns and relationships within the data. This process opens up the opportunity to retrain the model using the updated dataset and evaluate its performance with the new configuration.

Row	TaskType	UniqueTeams	AverageTimeToCompleteTheTask
1	Bike repair	Michael O'Neil	0.47775309454733866
2	Bike repair	Tina O'Neil	0.49394410827487095
3	Bike repair	Craig Stevenson	0.47442436744614913
4	Bike repair	John Grenelli	0.41014388511246208
5	Bike repair	Martha Williams	0.40756164859843136
6	Bike repair	Jack Myers	0.48080864720687655
7	Engine check	Michael O'Neil	1.2151707238839595
8	Engine check	Tina O'Neil	1.4821666666666666
9	Engine check	Jack Jackson	1.2414257566928018
10	Engine check	Ava Jones	1.4521153202143768
11	Engine check	John Grenelli	1.1900247260564161
12	Engine check	Michael Jackson	1.22253988031369
13	Engine check	Craig Brown	1.4224811385932077
14	Electronics repair	Jack Stevenson	3.2706209215167554
15	Electronics repair	Martha Johnson	3.236213856713857

Figure 18: Modified Dataset for ML model Version 2

Train and Test sets

We saved the result into a table *MwwMsSAPData.Inputdata_train2* so that we can use it for model training. The dataset was split into train and test sets as follows:

Query used:

```
-- Split the dataset into a training set and a test set
CREATE OR REPLACE TABLE MwwMsSAPData.Inputdata_train2 AS
SELECT TaskType,
UniqueTeams,
AverageTimeToCompleteTheTask
FROM MwwMsSAPData.Inputdata
WHERE MOD(ABS(FARM_FINGERPRINT(TeamId)), 10) < 8; -- 80% for training
CREATE OR REPLACE TABLE MwwMsSAPData.Inputdata_test2 AS
SELECT
TaskType,
UniqueTeams,
AverageTimeToCompleteTheTask
FROM MwwMsSAPData.Inputdata
WHERE MOD(ABS(FARM_FINGERPRINT(TeamId)), 10) >= 8; -- 20% for testing
```

5.1.2.2 ML model

Model Creation

Query used:

```sql
-- Train the regression model
CREATE OR REPLACE MODEL MwwMsSAPData.TimePredictionModelwithoutteamid
OPTIONS(model_type='linear_reg') AS
SELECT
  TaskType,
  UniqueTeams,
  AverageTimeToCompleteTheTask AS label
FROM MwwMsSAPData.Inputdata_train2;
```

Model Prediction

In the below diagram our models is predicting the best team or person available for certain task

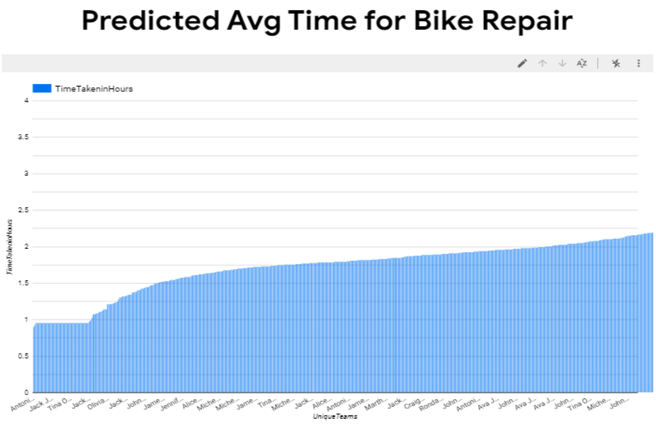


Figure 19: Model Prediction for Bike Repair

Model Evaluation

During the evaluation phase, the performance of the newly created model was assessed using the training and evaluation loss metrics. The training loss represents the discrepancy between the predicted values and the actual values in the training dataset, while the evaluation loss measures the performance of the model on unseen data.

The evaluation results showed a substantial improvement compared to the previous model. The training loss decreased to 0.051, indicating that the model was able to fit the training data more accurately and minimize the errors in predicting the AverageTimeToCompleteTheTask. The evaluation loss also improved significantly, reducing to 0.427. This suggests that the model's generalization capability improved, as it achieved lower loss on previously unseen data.

The model underwent 8 iterations during the training process. Each iteration involved updating the model's parameters based on the optimization algorithm, refining the model's predictions and reducing the loss. The iterative nature of the training process allowed the model to learn from the data and gradually improve its performance.

Overall, the improved model demonstrated better accuracy and predictive capability, as reflected in the reduced training and evaluation loss values. These results indicate that the refined feature space and the updated training approach contributed to enhancing the model's performance in predicting the time taken to complete tasks.

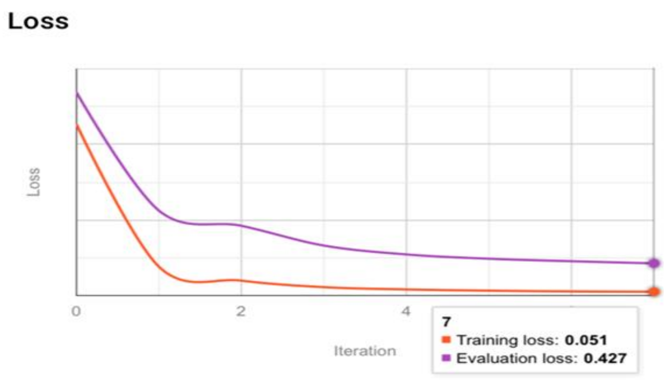


Figure 20 : Training & Evaluation Loss of ML model - version 2

5.2 Simulation Model

Tools used: BigQuery, App Engine

Github Repository - https://github.com/heyakshayjain/VLBA_OVGU_Project.git

Application URL: https://proj-iv-maintenance-service.uc.r.appspot.com

For creating the simulation model, we implemented a Flask application and deployed it on App Engine. The application has a home page and a result page. The home page has options to select the type of task for which the team prediction is required, the number of such tasks that need to be done, and the best n number of employees assignments for that task.

On the home page, we designed four drop-down menus to facilitate user input and configuration for the simulation.

Input:

- **ClientID-** specific client for which the task assignment will be simulated.
- **Task type-** nature of the task for which employees or teams will be assigned.
- **Quantity selection-** desired quantity or number of tasks to be assigned.
- **Best n-employees-** number of top-performing employees or teams to be considered for task assignments.

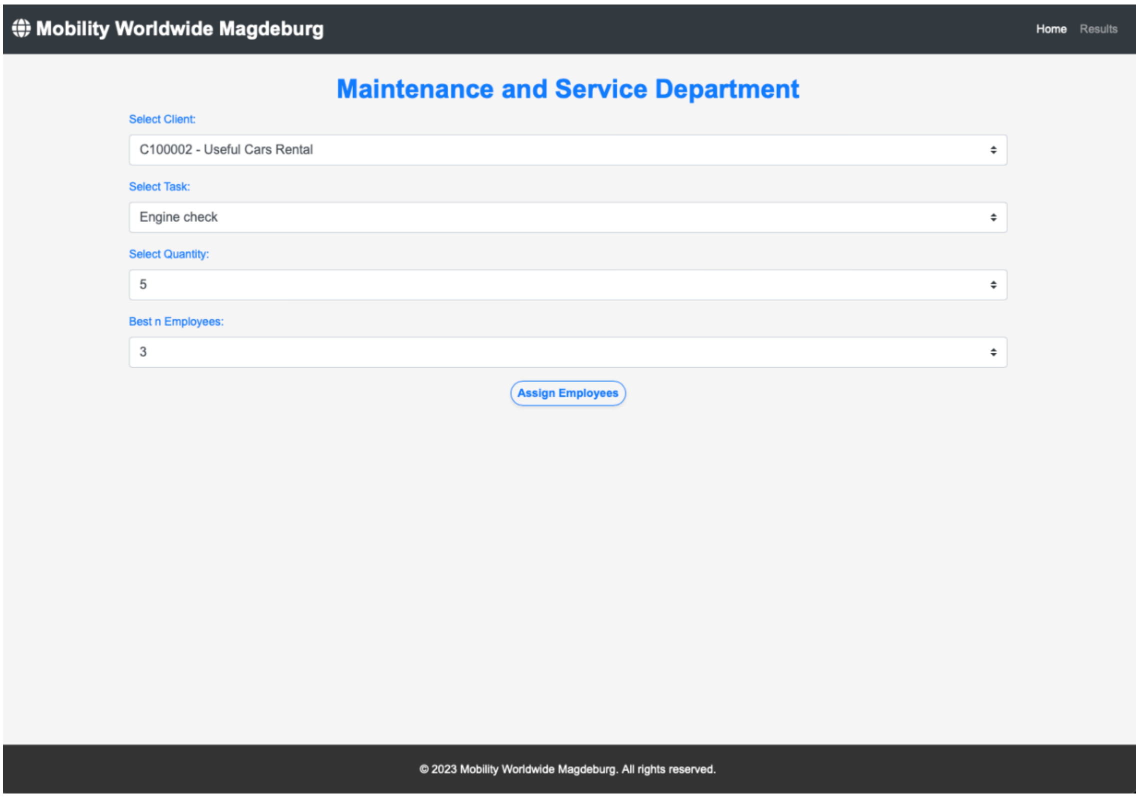


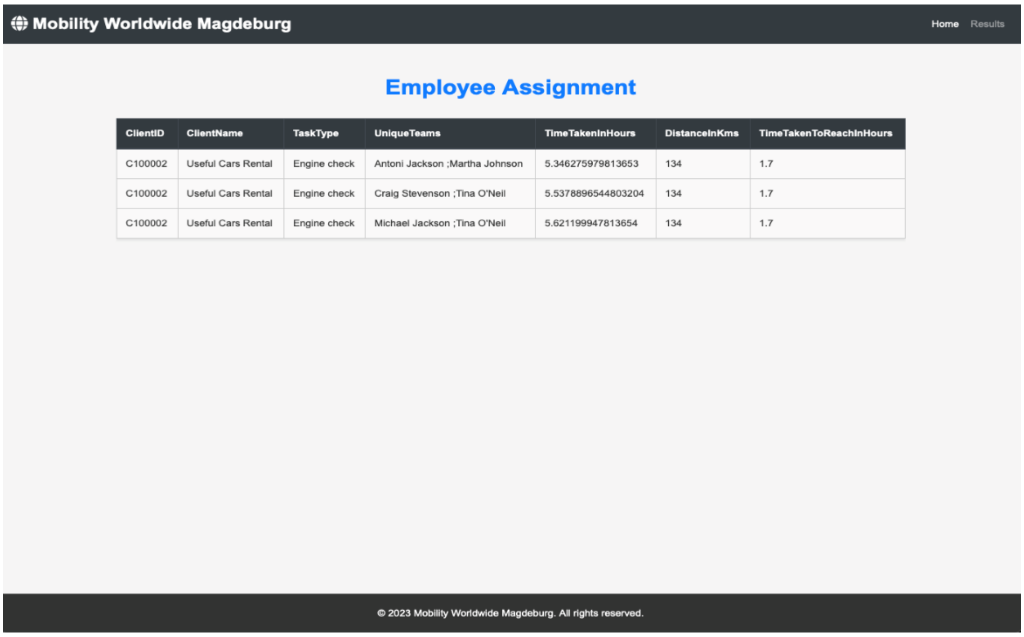
Figure 21: Application UI

Upon clicking the **"Assign Employees"** button on the simulation model, the application processes the inputs. It utilizes the machine learning model we created earlier to predict the time taken by each individual employee or team to complete the assigned task. To consider the Client distance from the employee to the Service Point, we have also added Client Name, Distance and Time Taken to Reach the site in our results. The predicted time estimates for all employees or teams are then arranged in ascending order, indicating the shortest to longest estimated completion times.

<u>Query Used:</u>

```
SELECT
d.ClientID,
d.ClientName,
'{selected_task}' AS TaskType,
i.UniqueTeams,
p.predicted_label*{selected_quantity} AS TimeTakenInHours,
d.Distance__km_ AS DistanceInKms,
d.Time__hrs_ AS TimeTakenToReachInHours
FROM (
SELECT DISTINCT UniqueTeams
FROM MwwMsSAPData.Inputdata_train2
) AS i
LEFT JOIN ML.PREDICT(MODEL `MwwMsSAPData.TimePredictionModelwithoutteamid`, (
```

```
SELECT
'{selected_task}' AS TaskType,
UniqueTeams
FROM (
SELECT DISTINCT UniqueTeams
FROM MwwMsSAPData.Inputdata_train2
)
)) AS p
ON i.UniqueTeams = p.UniqueTeams
CROSS JOIN proj-iv-maintenance-service.SampleTestingDataset.DistancesAndTimeServicePointToClient AS d
WHERE d.ClientID = '{selectedClient}'
ORDER BY p.predicted_label ASC;
```



ClientID	ClientName	TaskType	UniqueTeams	TimeTakenInHours	DistanceInKms	TimeTakenToReachInHours
C100002	Useful Cars Rental	Engine check	Antoni Jackson ;Martha Johnson	5.346275979813653	134	1.7
C100002	Useful Cars Rental	Engine check	Craig Stevenson ;Tina O'Neil	5.5378896544803204	134	1.7
C100002	Useful Cars Rental	Engine check	Michael Jackson ;Tina O'Neil	5.621199947813654	134	1.7

Figure 22: Team Prediction for Engine Check - Task Qty=5, No. of Teams=3

Choosing App Engine over Computer Engine to deploy our application

While we had multiple options through which we could have deployed our application, we chose App Engine. Our initial thought was to choose Compute engine but App Engine has multiple advantages over the former.

Compute Engine is an Infrastructure as a Service while App Engine is a Platform as a Service. So, we have to manage everything on the Compute engine and the App Engine is fully managed by Google. App Engine has a very efficient autoscaling which helps it to scale down the instances to zero when no requests are coming. Not to mention it is more secure than Compute Engine

6. System Architecture

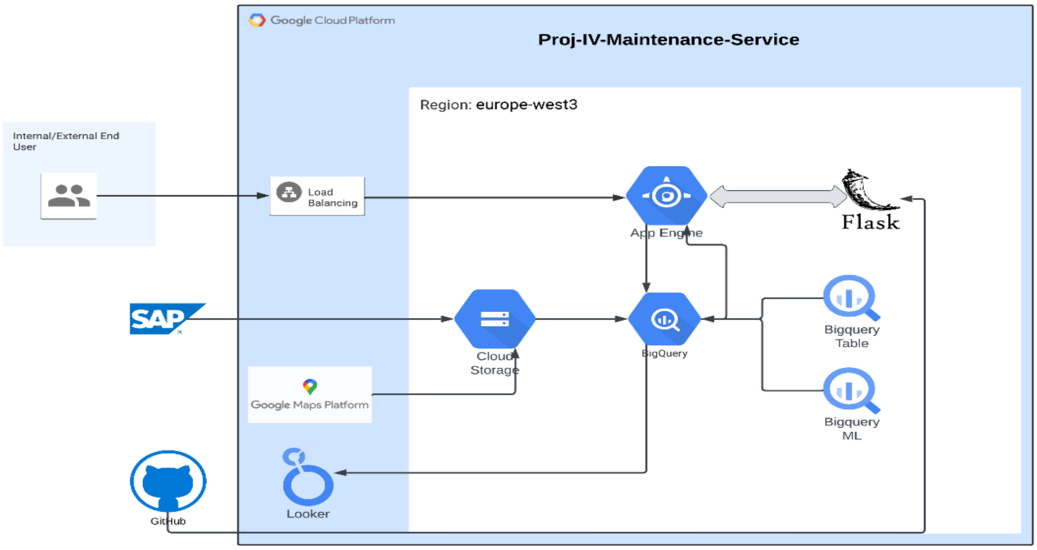


Figure 23: Architecture of our Project System

Main Components:

- **Cloud storage**: RESTful online file storage web service for storing and accessing data on Google Cloud Platform infrastructure. The crucial data including the SAP warehouse data provided by MWW and the clinet location distance information are stored in the google cloud storage.

- **BigQuery:** Google's fully managed, serverless data warehouse that enables scalable analysis over petabytes of data. It is a Platform as a Service (PaaS) that supports querying using a dialect of SQL. The datasets were stored in bigquery tables and processed with bigqueryML to perform the analysis needed for the tasks. The Looker Studio visualizations also fetch the data from BigQuery.

- **App Engine:** It is a fully managed, serverless platform for developing and hosting web applications at scale. We developed a Flask application to predict teams for new tasks. The application is hosted on app engine. The application interacts with bigquery tables for prediction of teams. App engine handles the provisioning of servers and scaling for the application.

7. Conclusion

The aim of this project was to increase effectiveness of service delivery for MWW by optimizing employee assignments and analyzing the learning curves of individuals and teams over time. We were able to achieve these objectives by utilizing the internal business data provided by MWW and by developing a simulation for predicting best employee assignments.

Through the analysis of the historical service data, we could identify patterns and extract valuable insights regarding employee performance for different type of tasks. This enabled us to identify the best-performing individuals and teams on the basis of their service records. This project's outcomes consist of analysis results, recommendations, and visualizations to support the decision-making processes for employee assignments. MWW can increase their service delivery efficiency by using the simulation developed in this project. The results of this project serve as a foundation for continuous improvement, enabling MWW to adapt to evolving customer needs and organizational growth.

In conclusion, this project enabled us to come up with meaningful insights and tools for MWW to optimize the service delivery processes, leverage employee expertise, and enhance customer satisfaction. With the help of data-driven decision-making and continuous improvement, MWW will be able to excel in provisioning high-quality services and meet the evolving demands of their customers.