Applying Machine Learning in Stand_up Data for Company Satalia

Yingji Diao

A Thesis Presented for the Degree of

Master of Science

Supervisor:
Dr. Daniel Hulme

Department of Computer Sciences
University College London
London, UK

September, 2018

Applying Machine Learning in Stand_up Data for Company Satalia

Abstract

Nowadays, in many companies and organizations, managers often have to face a common challenge. That is, how to manage the company so that the manager can have a clear understanding about the company's operation status to make sure the company are developing in a right direction. Satalia, as a technique corporation in England, provides AI solutions for companies and industries. Different from many other companies where there are managers to manage the organisation, Satalia uses AI technique to manage the company. In order to know the working status of every employee, a platform 'Hubble' is built to record what a certain employee has done in one day. This recorded information is called 'stand up'. However, at present, there is a problem in the company. Employees in Satalia are not active to complete their 'stand ups' since the process is purely manually and some employees often forget or are not patient to do this, which may make the company operate in a low efficiency and lead to a risk for the company. Based on this, this project finds a way to solve this issue by making a prediction on the 'stand ups'. Firstly, we use multiple neural network and Gaussian Naive Bayesian model to solve a classification problem concerning what projects a specific employee will work on in one day. Then, polynomial regression, random forest regression and Support vector machine (SVM) regression are used to make a prediction on the time that an employee spent in one project in a certain day. The research results show that in general, for the classification problem, the multiple neural network algorithm can get the best result with the label powerset method. For the prediction on 'time', the random forest regression is the best. Also, some further works of this project are also disscussed.

Acknowledgements

Firstly, I would like to present my great thanks to my supervisor Dr. Daniel Hulme who gave me valuable advice in the whole master dissertation process. I would also like to thank Gerard Cardoso Negrie from Satalia who gave me encouragement and helpful guidance when I encounter challenges and problems.

Working under their supervision was such a good experience and also provided me with many skills that are important in order to contribute effectively in a research process, and also a foundation into the next stage of my career.

Besides, I want to thank my partner Yujie Huang for his help and suggestions. Additionally, appreciations to University College London, which has helped me foster a lot of essential skills needed to excel in my master studying and career path. Thanks to my parents and all my friends for always helping and reminding about my project structure, schedule and deadlines, making sure that I was progressing as planned.

Contents

Chapter 1

Introduction

This chapter is the introduction part for this project. It makes a description of the background information and the main objectives that have been tried to achieve within this work. Also, an introduction of dataset is given. Besides, this chapter describes the structure of the dissertation which gives a brief outline for each chapter.

1.1 Background Information

Artificial Intelligence (AI), compared with the natural intelligence displayed by humans and other animals, is the intelligence demonstrated with machines [8]. Machine learning is a subfield of AI where statistical techniques are applied to make computer systems have the capability to learn with data, without being explicitly programmed [22]. Machine learning has many applications in different fields such as agriculture, marketing, insurance, financial market analysis and so on. In the past decades, machine learning has been widely used in many companies to solve the challenging problems in the company [37]. For instance, in a corporation, managers should make a strategy so that they can have a clear control on the company to make sure the company is operating in a right direction. In that case, the IT department in the company is often required by the management team to apply machine learning techniques to solve some challenging problems. There is an example in the company Satalia. As a technique corporation in England,

Satalia provides AI solutions for companies and industries. The working content of this company is nearly all related to different projects undertaken by employees with different skills. It is a flat, non-hierarchical company, which means that there are no manager existing within the organization. With the aim of continue operating as a non-hierarchical company, Satalia uses data science technique to manage the organization. That is, they try to understand the nature of activities that are occurring within the organization by automating as many manual processes as possible. There are basically five tools which are used to collect data information from employees to make a data analysis: Google for productivity tools, Slack for communications in the company environment, Github for software development projects, Atlassian for project management and an internal developed information system called Hubble, which records the stand_up information of employees including the id number of employees, circles(business function in Satalia), projects and interest groups(skills used in project). At present, the company faces a problem: the 'stand_ups' data is always not completed because the recording process is purely manually and some employees often forget or are not patient to do this. In that circumstance,the organization cannot get the complete working information from the employees, which makes the company operate with a low efficiency and may lead to a risk for the company.

1.2 Project Objective

Based on this problem, in order to increase the working efficiency and avoid the risk, this project tries to solve this issue by making a prediction on the 'stand_ups' data. Firstly, we solve a classification problem on what projects an employee has done in one certain day by using multiple neural networks and Gaussian Naive Bayesian model. Then, we use the polynomial regression, random forest regression and Support vector machine (SVM) regression to make a regression prediction on the 'time' which is the time duration an employee spent on a project. For these two predictions, We want to know which machine learning model is better fitted the 'stand_up' data with a higher accuracy. Also, we want to check whether

the results of the project can be used for Satalia.

1.3 Dataset Description

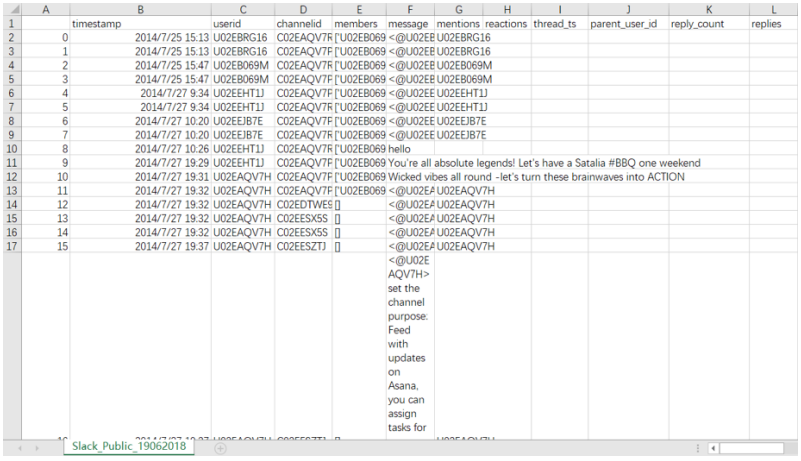
The dataset are provided by Satalia, which contain the general information about the project of the employees in the company. The dataset consist of six files: 'all_standups', 'Slack_Public' 'Slack_Private', 'User_mapping'. 'Interest_group_info' and 'Channel_info' respectively. The first dataset 'all_standups' makes a description on the 'stand_ups' data. From figure 1.1, we can see that an employee can have multiple projects in one day. There are totally six different columns. 'comment' is a self-description of the working content from an employee. The skills used in this project are described by the column 'interest_name' and 'interest_id' together. 'project_id' represents a specific project with its corresponding ID number. The fifth column 'standup_date' records the date when a certain employee did the project. The time length that an employee takes is described by the sixth column 'time'. 'user-id' represents a specific employee with its corresponding ID number. All the information described above shows in figure1.1

	A	B	C	D	E	F	G
1	comment	interest_id	interest_name	project_id	standup_date	time	user_id
2	AV Demo	6	_comms_infrastructure	70	2017/6/23 0:00	1:36	56
3	Met with	4	_comms_information	70	2017/7/4 0:00	2:00	11
4	Met with	4	_comms_information	70	2017/7/4 0:00	2:40	141
5	Looked at	5	_comms_knowledge	70	2017/7/5 0:00	8:00	141
6	Attended	5	_comms_knowledge	70	2017/7/7 0:00	2:00	85
7	Cleaned u	29	_x_navigation	70	2017/6/22 0:00	2:40	3
8	Researche	4	_comms_information	70	2017/7/10 0:00	1:20	141
9	Did lunch	5	_comms_knowledge	70	2017/7/7 0:00	8:00	16
10	Went to B	5	_comms_knowledge	70	2017/7/10 0:00	4:00	147
11	Went to B	25	_connect_media	70	2017/7/10 0:00	4:00	147
12	Read past	4	_comms_information	70	2017/7/11 0:00	2:40	141
13	Worked o	4	_comms_information	70	2017/7/12 0:00	4:00	141
14	Attended	5	_comms_knowledge	70	2017/7/28 0:00	2:40	79
15	Worked o	4	_comms_information	70	2017/7/14 0:00	4:00	141
16	Finished a	4	_comms_information	70	2017/7/17 0:00	2:00	141
17	Attended	6	_comms_infrastructure	70	2017/7/7 0:00	1:20	56
18	Attended	4	_comms_information	70	2017/7/19 0:00	1:08	11
19	Looking a	4	_comms_information	70	2017/7/22 0:00	1:36	4
20	Met with	29	_x_navigation	70	2017/7/24 0:00	2:40	141
21	Patent dis	3	_comms_security	70	2017/7/24 0:00	2:40	81
22	Hosted Tu	29	_x_navigation	70	2017/7/26 0:00	2:00	74
23	Attended	5	_comms_knowledge	70	2017/7/26 0:00	1:20	14
24	Attended	5	_comms_knowledge	70	2017/7/26 0:00	4:00	141
25	Catch up	6	_comms_infrastructure	70	2017/7/27 0:00	2:40	56
26	Attended	4	_comms_information	70	2017/7/26 0:00	2:00	83
27	Attended	5	_comms_knowledge	70	2017/7/26 0:00	2:40	79
28	Created n	5	_comms_knowledge	70	2017/7/31 0:00	4:00	74
29	Added ins	5	_comms_knowledge	70	2017/7/31 0:00	2:40	79
30	Looked in	3	_comms_security	70	2017/7/17 0:00	1:36	3

all_standups_jun17_jun18

Figure 1.1: Stand_up Data

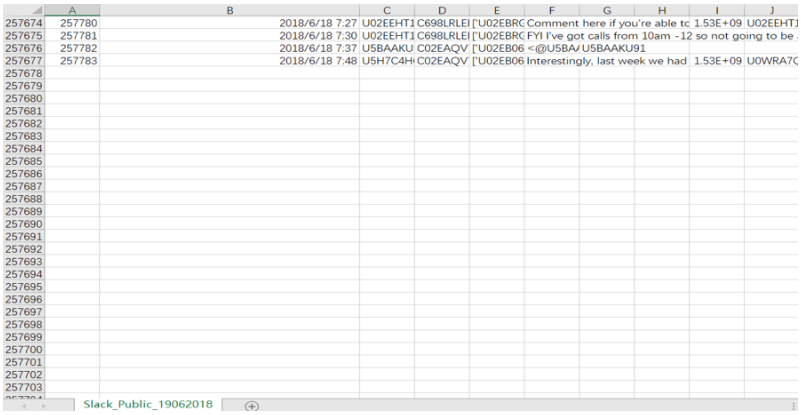
'Slack_Public' is the second dataset which contains the communication information in the entire environment of the company through the Slack platform. As shown in figure 1.2 and 1.3 , There are total 257782 rows of information. The second column 'timestamp' denotes the date from the date of 2014/7/25 to 2018/6/18. The third column 'user_id' is presented in the character format which is different from the digital forms as shown in 'all_standups'. Data of 'channelid' is from the dataset 'channel_info' with some mappings to 'project_id'. The fifth column 'members' makes a description of who are related to the talk. The information of communication content is presented in the column 'message'. The column 'mentions' is the mention of an employee in the communication process. The data 'reactions' is the reply from one employee.



	A	timestamp	userid	channelid	members	message	mentions	reactions	thread_ts	parent_user_id	reply_count	replies
2	0	2014/7/25 15:13	U02EBRG16	C02EAQV7F	['U02EB069	<@U02EE	U02EBRG16					
3	1	2014/7/25 15:13	U02EBRG16	C02EAQV7F	['U02EB069	<@U02EE	U02EBRG16					
4	2	2014/7/25 15:47	U02EB069M	C02EAQV7F	['U02EB069	<@U02EE	U02EB069M					
5	3	2014/7/25 15:47	U02EB069M	C02EAQV7F	['U02EB069	<@U02EE	U02EB069M					
6	4	2014/7/27 9:34	U02EEHT1J	C02EAQV7F	['U02EB069	<@U02EE	U02EEHT1J					
7	5	2014/7/27 9:34	U02EEHT1J	C02EAQV7F	['U02EB069	<@U02EE	U02EEHT1J					
8	6	2014/7/27 10:20	U02EEJB7E	C02EAQV7F	['U02EB069	<@U02EE	U02EEJB7E					
9	7	2014/7/27 10:20	U02EEJB7E	C02EAQV7F	['U02EB069	<@U02EE	U02EEJB7E					
10	8	2014/7/27 10:26	U02EEHT1J	C02EAQV7F	['U02EB069	hello						
11	9	2014/7/27 19:29	U02EEHT1J	C02EAQV7F	['U02EB069	You're all absolute legends! Let's have a Satalia #BBQ one weekend						
12	10	2014/7/27 19:31	U02EAQV7H	C02EAQV7F	['U02EB069	Wicked vibes all round -let's turn these brainwaves into ACTION						
13	11	2014/7/27 19:32	U02EAQV7H	C02EAQV7F	['U02EB069	<@U02EA	U02EAQV7H					
14	12	2014/7/27 19:32	U02EAQV7H	C02EDTWES	[]	<@U02EA	U02EAQV7H					
15	13	2014/7/27 19:32	U02EAQV7H	C02EESX5S	[]	<@U02EA	U02EAQV7H					
16	14	2014/7/27 19:32	U02EAQV7H	C02EESX5S	[]	<@U02EA	U02EAQV7H					
17	15	2014/7/27 19:37	U02EAQV7H	C02EESZTJ	[]	<@U02EA	U02EAQV7H					
						<@U02E AQV7H> set the channel purpose: Feed with updates on Asana, you can assign tasks for						

Slack_Public_19062018

Figure 1.2: Slack_Public Data

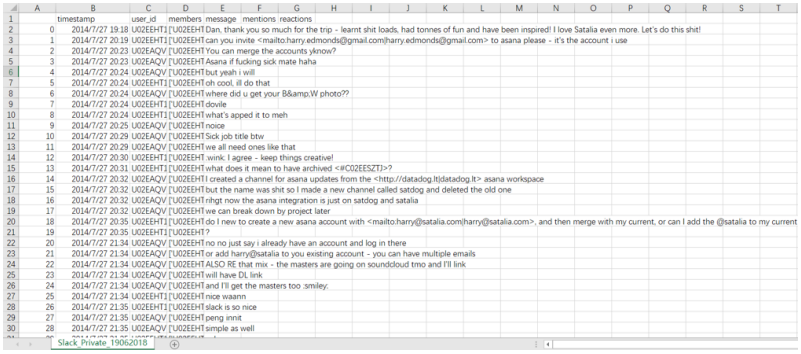


	A	B	C	D	E	F	G	H	I	J
257674	257780	2018/6/18 7:27	U02EEHT1	C698LRLEI	['U02EBRC	Comment here if you're able tc			1.53E+09	U02EEHT1
257675	257781	2018/6/18 7:30	U02EEHT1	C698LRLEI	['U02EBRC	FYI I've got calls from 10am -12 so not going to be				
257676	257782	2018/6/18 7:37	U5BAAKU	C02EAQV	['U02EB06	<@U5BA/ U5BAAKU91				
257677	257783	2018/6/18 7:48	U5H7C4H	C02EAQV	['U02EB06	Interestingly, last week we had			1.53E+09	U0WRA7
257678										
257679										
257680										
257681										
257682										
257683										
257684										
257685										
257686										
257687										
257688										
257689										
257690										
257691										
257692										
257693										
257694										
257695										
257696										
257697										
257698										
257699										
257700										
257701										
257702										
257703										

Slack_Public_19062018

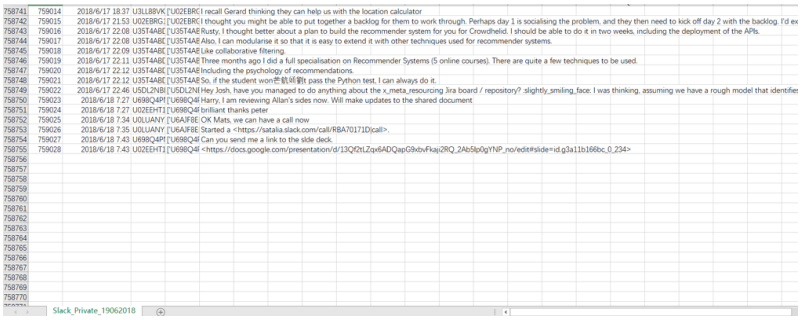
Figure 1.3: Slack_Public Data

'Slack_Private' is the third dataset (figure 1.4) which makes a description of the communication process between one employee and some other employees in the Slack platform with the form of Private. There are total 759027 rows of information. From figure 1.4 and 1.5, the date is from 2014/7/27 to 2018/6/18 which is recorded in the 'timestamp' column. The meanings of the column 'user_id', 'members', 'message', 'mentions' and 'reactions' are the same as the 'Slack_Public'.



	timestamp	user_id	members	message	mentions	reactions
0	2014/7/27 19:18	U02EEHTJ	['U02EEHTJ']	Dan, thank you so much for the trip - learnt shit loads, had tonnes of fun and have been inspired! I love Satalia even more. Let's do this shit!		
1	2014/7/27 20:19	U02EEHTJ	['U02EEHTJ']	can you invite <mailto:harry.edmonds@gmail.com|harry.edmonds@gmail.com> to asana please - it's the account i use		
2	2014/7/27 20:23	U02EAQV	['U02EEHTJ']	You can merge the accounts yknow?		
3	2014/7/27 20:23	U02EAQV	['U02EEHTJ']	Asana if fucking sick mate haha		
4	2014/7/27 20:24	U02EAQV	['U02EEHTJ']	but yeah i will		
5	2014/7/27 20:24	U02EEHTJ	['U02EEHTJ']	oh cool, ill do that		
6	2014/7/27 20:24	U02EAQV	['U02EEHTJ']	where did u get your B&W photo??		
7	2014/7/27 20:24	U02EEHTJ	['U02EEHTJ']	dovile		
8	2014/7/27 20:24	U02EEHTJ	['U02EEHTJ']	what's apped it to meh		
9	2014/7/27 20:25	U02EAQV	['U02EEHTJ']	noice		
10	2014/7/27 20:29	U02EAQV	['U02EEHTJ']	Sick job title btw		
11	2014/7/27 20:29	U02EAQV	['U02EEHTJ']	we all need ones like that		
12	2014/7/27 20:30	U02EEHTJ	['U02EEHTJ']	:wink: I agree - keep things creative!		
13	2014/7/27 20:31	U02EEHTJ	['U02EEHTJ']	what does it mean to have archived <#C02EEEZT1>?		
14	2014/7/27 20:32	U02EAQV	['U02EEHTJ']	I created a channel for asana updates from the <http://datadog.lt|datadog.lt> asana workspace		
15	2014/7/27 20:32	U02EAQV	['U02EEHTJ']	but the name was shit so I made a new channel called satdog and deleted the old one		
16	2014/7/27 20:32	U02EAQV	['U02EEHTJ']	rihgt now the asana integration is just on satdog and satalia		
17	2014/7/27 20:32	U02EAQV	['U02EEHTJ']	we can break down by project later		
18	2014/7/27 20:35	U02EEHTJ	['U02EEHTJ']	do I new to create a new asana account with <mailto:harry@satalia.com|harry@satalia.com>, and then merge with my current, or can I add the @satalia to my current		
19	2014/7/27 20:35	U02EEHTJ	['U02EEHTJ']	?		
20	2014/7/27 21:34	U02EAQV	['U02EEHTJ']	no no just say i already have an account and log in there		
21	2014/7/27 21:34	U02EAQV	['U02EEHTJ']	or add harry@satalia to you existing account - you can have multiple emails		
22	2014/7/27 21:34	U02EAQV	['U02EEHTJ']	ALSO RE that mix - the masters are going on soundcloud tmo and I'll link		
23	2014/7/27 21:34	U02EAQV	['U02EEHTJ']	will have DL link		
24	2014/7/27 21:34	U02EAQV	['U02EEHTJ']	and I'll get the masters too :smiley:		
25	2014/7/27 21:34	U02EEHTJ	['U02EEHTJ']	nice waann		
26	2014/7/27 21:35	U02EEHTJ	['U02EEHTJ']	slack is so nice		
27	2014/7/27 21:35	U02EAQV	['U02EEHTJ']	peng innit		
28	2014/7/27 21:35	U02EAQV	['U02EEHTJ']	simple as well		

Figure 1.4: Slack_Private Data Start

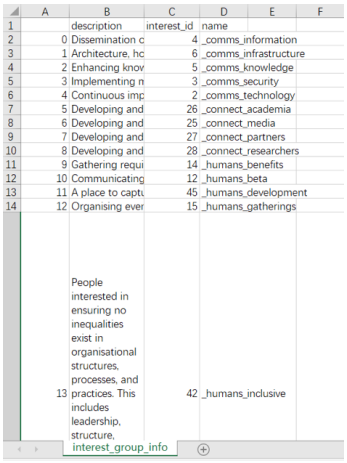


758741	759014	2018/6/17 18:37	U3LL88VK	['U02EBRC']	recall Gerard thinking they can help us with the location calculator	
758742	759015	2018/6/17 21:53	U02EBRC	['U02EBRC']	thought you might be able to put together a backlog for them to work through. Perhaps day 1 is socialising the problem, and they then need to kick off day 2 with the backlog. I'd ex	
758743	759016	2018/6/17 22:08	U35T4ABE	['U35T4ABE']	Rusty, I thought better about a plan to build the recommender system for you for Crowdhelid. I should be able to do it in two weeks, including the deployment of the APIs.	
758744	759017	2018/6/17 22:08	U35T4ABE	['U35T4ABE']	Also, I can modularise it so that it is easy to extend it with other techniques used for recommender systems.	
758745	759018	2018/6/17 22:09	U35T4ABE	['U35T4ABE']	Like collaborative filtering.	
758746	759019	2018/6/17 22:11	U35T4ABE	['U35T4ABE']	Three months ago I did a full specialisation on Recommender Systems (5 online courses). There are quite a few techniques to be used.	
758747	759020	2018/6/17 22:12	U35T4ABE	['U35T4ABE']	Including the psychology of recommendations.	
758748	759021	2018/6/17 22:12	U35T4ABE	['U35T4ABE']	So, if the student won't pass the Python test, I can always do it.	
758749	759022	2018/6/17 22:46	U5DLZNB	['U5DLZNB']	Hey Josh, have you managed to do anything about the x_meta_resourcing Jira board / repository? :slightly_smiling_face: I was thinking, assuming we have a rough model that identifies	
758750	759023	2018/6/18 7:27	U698Q4P	['U698Q4P']	Harry, I am reviewing Allan's sides now. Will make updates to the shared document	
758751	759024	2018/6/18 7:27	U02EEHTJ	['U698Q4P']	brilliant thanks peter	
758752	759025	2018/6/18 7:34	U0LUANY	['U6AJF8E']	OK Mats, we can have a call now	
758753	759026	2018/6/18 7:35	U0LUANY	['U6AJF8E']	Started a <https://satalia.slack.com/call/R8A70171D|call>.	
758754	759027	2018/6/18 7:43	U698Q4P	['U698Q4P']	Can you send me a link to the slde deck.	
758755	759028	2018/6/18 7:43	U02EEHTJ	['U698Q4P']	<https://docs.google.com/presentation/d/13QfZtLZqvl6ADQapG9xbvFkajv2RQ_2Ab5lp0gYNP_no/edit#slide=id.g3a11b166bc_0_234>	

Figure 1.5: Slack_Private Data End

The fifth file 'Interest_group_info' (figure 1.6) makes a description of the skills used by employees in their projects. The second column 'description' makes a full explanation for each skill. The third and fourth columns together present a mapping between 'interest_id' and 'name'.

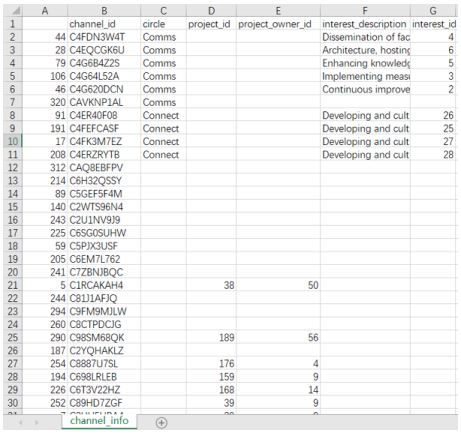
For the last file 'Channel_info' (figure 1.7), it presents a link between the 'channel_id' and 'project_id'. The third column 'circle' describes the department of the company. The fifth column 'project_owner_id' denotes the 'user_id' according to the 'project_id'.



	description	interest_id	name
0	Dissemination c	4	_comms_information
1	Architecture, hc	6	_comms_infrastructure
2	Enhancing knov	5	_comms_knowledge
3	Implementing n	3	_comms_security
4	Continuous imp	2	_comms_technology
5	Developing and	26	_connect_academia
6	Developing and	25	_connect_media
7	Developing and	27	_connect_partners
8	Developing and	28	_connect_researchers
9	Gathering requi	14	_humans_benefits
10	Communicating	12	_humans_beta
11	A place to capt	45	_humans_development
12	Organising ever	15	_humans_gatherings
13	People interested in ensuring no inequalities exist in organisational structures, processes, and practices. This includes leadership, structure,	42	_humans_inclusive

interest_group_info

Figure 1.6: Interest_Group_Info Data



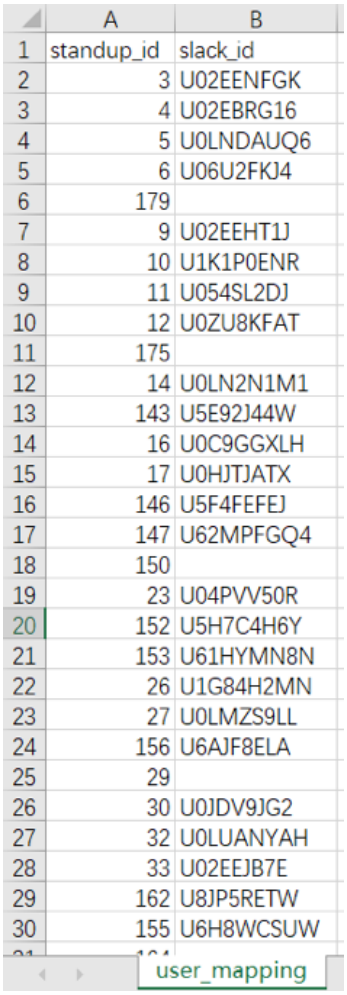
	channel_id	circle	project_id	project_owner_id	interest_description	interest_id
44	C4FDN3W4T	Comms			Dissemination of fac	4
28	C4EQCGK6U	Comms			Architecture, hosting	6
79	C4G6B4ZZS	Comms			Enhancing knowledg	5
106	C4G64L52A	Comms			Implementing measi	3
46	C4G620DCN	Comms			Continuous improve	2
320	CAVKNP1AL	Comms				
91	C4ER40F08	Connect			Developing and cult	26
191	C4FEFCASF	Connect			Developing and cult	25
17	C4FK3M7EZ	Connect			Developing and cult	27
208	C4ERZRYTB	Connect			Developing and cult	28
312	CAQ8EBFPV					
214	C6H32QSSY					
89	C5GEF5F4M					
140	C2WTS96N4					
243	C2U1NV9J9					
225	C6SG0SUHW					
59	C5PJX3USF					
205	C6EM7L762					
241	C7ZBNJBQC					
5	C1RCAKAH4		38	50		
244	C81J1AFJQ					
294	C9FM9MJLW					
260	C8CTPDCJG					
290	C98SM68QK		189	56		
187	C2YQHAKLZ					
254	C8887U7SL		176	4		
194	C698LRLEB		159	9		
226	C6T3V22HZ		168	14		
252	C89HD7ZGF		39	9		

channel_info

Figure 1.7: Channel_Info Data

For the fourth file ‘User_mapping’ (figure 1.8), it shows the relationship between the ‘standup_id’ and ‘slack_id’. The ‘standup_id’ is just another presentation of ‘user_id’.

All the three datasets ‘User_mapping’, ‘Interest_group_info’ and ‘Channel_info’ together make a connection between the dataset ‘all_standups’ and ‘Slack’ datset (‘Slack_Public’ and ‘Slack_Private’).



	A	B
1	standup_id	slack_id
2	3	U02EENFGK
3	4	U02EBRG16
4	5	U0LNDAUQ6
5	6	U06U2FKJ4
6	179	
7	9	U02EEHT1J
8	10	U1K1P0ENR
9	11	U054SL2DJ
10	12	U0ZU8KFAT
11	175	
12	14	U0LN2N1M1
13	143	U5E92J44W
14	16	U0C9GGXLH
15	17	U0HJTJATX
16	146	U5F4FEFEJ
17	147	U62MPFGQ4
18	150	
19	23	U04PVV50R
20	152	U5H7C4H6Y
21	153	U61HYMN8N
22	26	U1G84H2MN
23	27	U0LMZS9LL
24	156	U6AJF8ELA
25	29	
26	30	U0JDV9JG2
27	32	U0LUANYAH
28	33	U02EEJB7E
29	162	U8JP5RETW
30	155	U6H8WCSUW

user_mapping

Figure 1.8: User_mapping Data

1.4 Outlines

Here is the construction for this report:

- Chapter 1 Introduction - a brief introduction of the project
- Chapter 2 Literature review - will make an introduction on the theoretical reasoning for this project. Some basic principles and knowledge of machine learning will be discussed. Also, we discuss some basic knowledge of the machine learning models used in this project.
- Chapter 3 Experimental procedures - will make a detailed description of the experimental procedures containing data preprocessing, methodologies and the steps in experiment.

- Chapter 4 Results - will present the results obtained from different machine learning models and make a evaluation on these results.
- Chapter 5 Conclusion and Further work - will make a summary of this project and discuss some limitations and further works of our research.

Chapter 2

Literature review

In this chapter, some theories, principles and knowledge of machine learning from literature are discussed. they can be used as a guidance for this research project.

2.1 Basic Knowledge of Machine Learning

Machine learning is a subfield of Artificial Intelligence, as a statistical technique, it automates analytical model building to make the computer systems have the capability to "learn" with data [22]. It proceeds towards learning to solve problems by itself rather than pushing the commands by the programmer [6]. A more formal and widely quoted definition of the machine learning algorithms is provided by Tom M. Mitchell: "A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P if its performance at tasks in T, as measured by P, improves with experience E [47]." In general, as shown in graph 2.1, machine learning algorithms can be broadly classified into three categories: Unsupervised Learning talked mainly in [26], Supervised Learning in [9] and Reinforcement Learning in [62] and [63].

2.1.1 Unsupervised Machine Learning Algorithm

Unsupervised machine learning is the algorithm where no labelled data is available for training [27]. In other words, the data are "unlabeled" (not categorized). As the basic algorithm, it only has input data with no corresponding output vari-

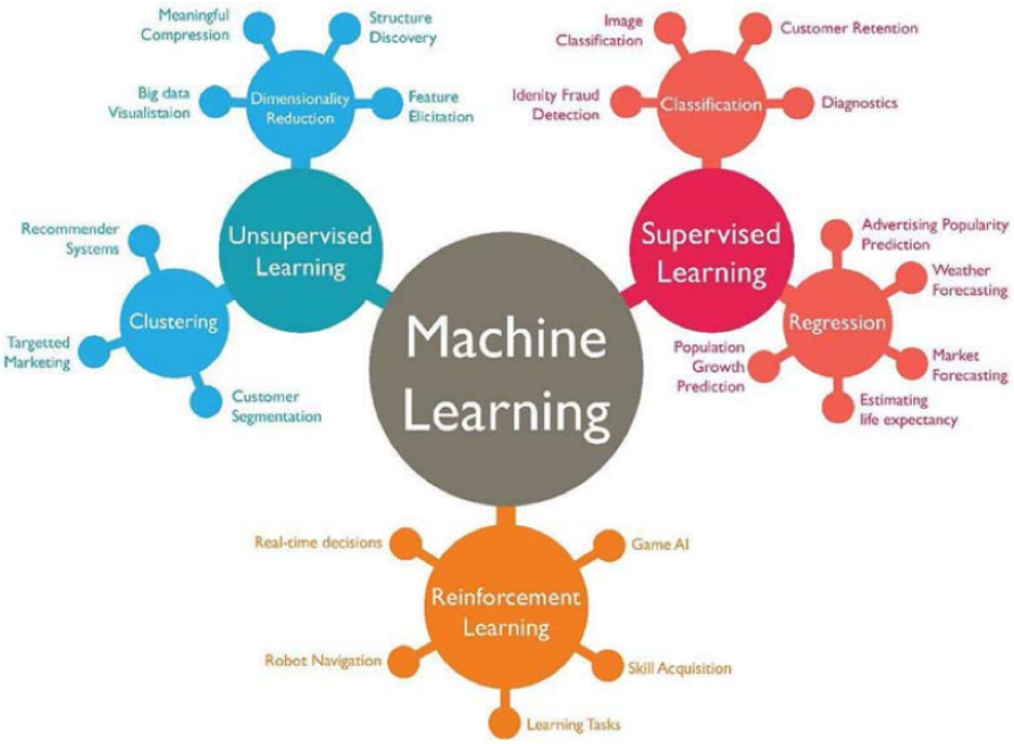


Figure 2.1: Machine Learning Description

ables since for a particular input there is no corresponding output [27][30]. In the algorithm, the data are arranged into a group of clusters which describe the structure and make a simplification on the complex data to make them organized for analysis [30]. The unsupervised learning problems can be generally classified into clustering which aims at discovering inherent grouping and problems of association [38]. Clustering (such as k-means) and Anomaly detection are the two commonly used unsupervised learning algorithms. There are some common applications of unsupervised learning such as targeting marketing, customer segmentation and dimensionality reduction in visualization of big data, k-means in recommender systems for clustering, feature elicitation, discovery of structure and others [38].

2.1.2 Supervised Machine Learning Algorithm

Different from unsupervised machine learning algorithm, supervised machine learning algorithm contains both inputs and corresponding outputs in the progress [57]. In other words, the data are 'labeled' in this algorithm. It focuses on mak-

ing predictions and searching for patterns on the given samples. The aim of this algorithm is trying to find the relationships between the target prediction output and the input features. In more detailed, Given the output variable (Y) and the input variables (X), we try to make a mapping from the input to the output and then build a function so that a relationship between output Y and input X are established which can then be utilized for prediction [34]. Generally, there are two types of supervised learning algorithms. The first one is regression, which is the problem of prediction on the relationship between an output variable (Y) and input variables (X) [34]. The other type is classification, which is the problem of classifying a observation based on a given data sample [34]. There are some common examples of supervised learning algorithms such as polynomial regression, random forest regression, Support Vector Machines (SVM) regression for regression problems and neural networks, random forest, K Nearest Neighbour (KNN), Support Vector Machines, Gaussian Naive Bayesian for classification problems [14]. Supervised learning is commonly used in regression problems such as life experience estimation, population growth prediction, weather forecasting and so on, and classification problems such as Diagnostics, Speech recognition, Digit recognition, Identity Fraud detection and others [48].

2.1.3 Reinforcement Machine Learning Algorithm

Reinforcement learning (RL), as a subfield of machine learning, focuses on the problem of what actions software agents should take to maximize the cumulative reward in an environment [71]. In other words, the observations collected from the interaction are utilized and actions are taken for maximizing the benefits and minimizing the risk. This algorithm forwards an action on the basis of the data point and later makes an assessment on the decision. Besides, it learns in an iterative fashion [4]. There are some common learning algorithms in the reinforcement such as Deep Adversarial Networks, Q-Learning, and Temporal Difference. Reinforcement learning Algorithm is widely applied in real-time decision, Game AI, learning tasks, skill acquisition and robot navigation [53].

2.2 Algorithm Selection

In this part, we decide on what machine learning algorithms should be utilized in our project based on the basic knowledge of machine learning. For all the six datasets 'all_standups', 'Slack_Public' 'Slack_Private', 'User_mapping'. 'Interest_group_info' and 'Channel_info', it is clear that they are 'labeled' data since the input and output information have been presented. That is, we should use supervised learning algorithms to solve the problems in this project.

2.2.1 Algorithm for Classification

2.2.1.1 Multiple Neural Networks

The multiple neural network algorithm consists of a network made up of the artificial neurons with complex global behavior, dependent on the connection between the elements for processing and parameters to solve AI problems [1]. It utilizes a computational or mathematical model for processing information based on a connectionist approach to make a computation with an interconnected group of natural or artificial neurons [72]. In most circumstances, as an adaptive system, multiple neural network make a change of its structure on the basis of the internal or external information. In practice, the multiple neural network is a tool for decision making or non-linear statistical data modeling. It can find patterns in data and model a complex input-output relationship [1] [72].

There are some applications of multiple neural network such as classification, regression analysis, function approximation, data processing, and others stated in [72]. There are some main advantages of multiple neural network algorithm [70]. Firstly, there are not many restrictions on the input variables in the multiple neural network algorithm concerning classification problems, which is different from many other prediction algorithms. Secondly, it can generalize after the learning process from the inputs, outputs and their relationships and thus can predict on unseen data.

2.2.1.2 Gaussian Naive Bayesian

In machine learning, Naive Bayes is a methodology that constructs classifiers which are the models that assign class labels drawn from some finite set to samples which are represented as feature values vectors [76]. It is an algorithm family on the basis of the principle: there is an assumption that, for all Naive Bayes classifiers, given the variable of class, the value of a particular feature is independent of the value of other features [31]. The Gaussian Naive Bayesian algorithm is the Naive Bayes when each class is distributed according to a Gaussian Distribution [76] [31]. In that case, Naive Bayes classifiers can be trained with high efficiency. Naive Bayes classifier has been proved to work significantly well in many complex situations in the real-world although it has an apparently oversimplified assumptions with a naive design [10]. In 2006, there was a comprehensive comparison between Bayes classification and other classification algorithms, the result showed that Naive Bayes outperformed other approaches, such as random forests in terms of multi-label classification [10]. Also, Naive Bayes has the advantage that only a small number of training data is required for the estimation of the classification parameters [52].

2.2.2 Algorithm for Regression

For the regression prediction, we use three regression models: polynomial, random forest and svm to predict the output 'time' with 'user_id', 'project_id' and 'interest_id' as inputs.

2.2.2.1 Polynomial Regression

Polynomial regression is the algorithm in the form of a n-th degree polynomial which focuses on finding the relationships between the input x and the output y [66]. Maximum Likelihood Estimation (MLE) is used in the parameter estimation to find the value of parameter[66]. It has a main advantage which has been proved by many literatures: the polynomial regression algorithm is more flexible since compared with many other machine learning algorithm, a wider range of

functions can be fitted [19].

2.2.2.2 Random Forest Regression

Random forest, as the extension from decision tree, is an easy-to-interpret and robust machine learning algorithm for regression and classification problems [40]. In regression, it is a fast way of learning a function that makes a mapping between output y and input x, where y is also numeric for regression and x is the numeric variable [40]. Although some algorithms such as Logistic Regression, SVM and Deep Neural Networks have the same basic principle with a advantage of addressing larger and more complicated dataset, they can take a lot of iterations and hyperparameter adjustments before a result is obtained and are extremely hard to interpret [15]. Furthermore, the most important advantage of utilizing random forest regression is that it is convenient for experimenter to see what variables or features are related to the regression and the relative importance on the basis of their location depth wise on the tree [61].

2.2.2.3 SVM Regression

Support vector machine (SVM), as a supervised learning algorithm, focuses on constructing a hyper-plane in a high-dimensional or infinite-dimensional space to make an analysis on the data used for classification and regression [11]. This algorithm tries to achieve a separation by the hyper-plane which has the largest distance to the nearest training-data point [60]. SVM has been proved by many literatures to have four main advantages: Firstly, it has a parameter on regularization to avoid the problem of over-fitting. Secondly, SVM is defined by a convex optimization problem and there are efficient methods to solve the problem. Thirdly, since the kernel trick is used in SVM, an expert knowledge of the problem via engineering the kernel can be built. Lastly, it is a bound approximation on the test error rate, which has been proved to be effective by a substantial theories [46].

2.3 Other Problems Involved

The multi-label classification problem is involved for this project. Besides, 'one-hot encoding' is used in the data preprocessing process. 'data normalization' technique is used during the experiment steps.

2.3.1 Multi-label Classification

Multi-Label Classification is a problem where an object is classified into one or more than two categories with a classifier trained for each label independently [56]. It is different from multi-class classification in which instances are only categorized into one of the classes. More specifically, the aim of the multi-label classification is to discover a model which relates the output vectors y (each element (label) y is assigned with 0 or 1) and the inputs x [56]. There are three commonly used problem transformation methods for simplifying multi-label classification problems to a single label problem: Binary relevance, Classifier Chain and Label Powerset [68]. Binary relevance (BR) method is the baseline method where one binary classifier is trained independently for each label. When an unseen sample is given, then all labels for this sample is predicted by the combined model when the respective classifiers predict a positive result [68].

The second problem transformation method is Label Powerset (LP) transformation where one binary classifier is created for every label combination in the training dataset. Label correlations is considered in this approach. More specifically, in a dataset, each different labels combination is regarded as a single label [13]. A single-label classifier $H : X \xrightarrow{p(L)}$ is trained after transformation, where for all labels in L, P(L) is the power set. This approach has a main drawback: as the number of labels increases, the number of label combinations grows exponentially, which significantly increases the classification running time [55].

Classifier Chains method is the third method which is a development from Binary relevance (BR) method and it is efficient even when there are a large number of labels. Besides, dependencies between labels are also considered. In other words, it has the computational efficiency of the BR method with the label depen-

dencies taken into consideration for classification [69].

2.3.2 One-hot Encoding

'One hot encoding' is a procedure in which a one-hot vector converted from categorical variables is provided to Machine learning algorithms for a better prediction. As a 1 × N matrix (vector), one-hot vector is utilized to differentiate each word from every other in a vocabulary [50]. The vector consists of 0s in all cells with the exception of a single 1 in a cell used uniquely to identify the word.

2.3.3 Data Normalization

In some machine learning algorithm process, the original data should be processed into zero-centered data and then normalized data [3]. The aim of normalization is adjusting original data values which are measured on different scales into a notionally common scale [5]. The whole process is shown in graph 2.2, the original data are firstly transformed into zero-centered data and then to the normalized data which follow a Gaussian distribution with a zero mean and standard deviation of one. It can be computed through formula , where X, u and σ is the value, the mean and the standard deviation of the original data respectively.

$$Z = \frac{(X - u)}{\sigma} \tag{2.3.1}$$

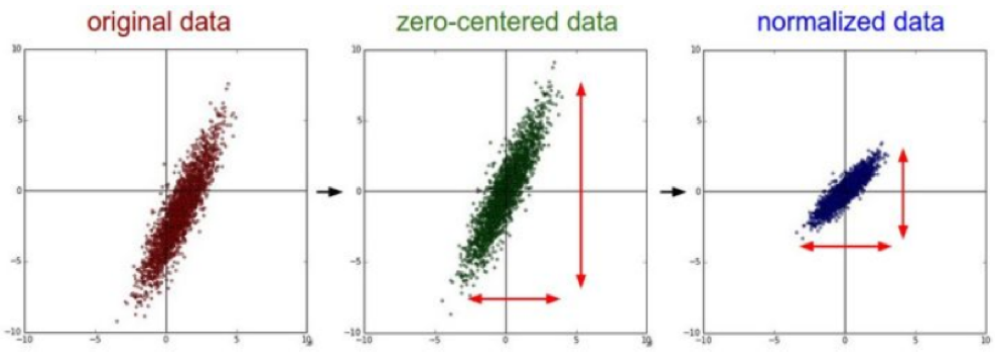


Figure 2.2: Data Normalization

2.3.4 Mean Squared Error

In mathematics, the mean squared error (MSE) makes a measure on the average of the errors squares which is the average squared difference between what is estimated and the estimated values. MSE is a measure on an estimator's quality. It is non-negative and the results are better when its values closer to zero [39].

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - y_i')^2 \tag{2.3.2}$$

Chapter 3

Experimental Procedures

3.1 Data Processing

Before the experiment, we make a data pre-processing since the information from all the six datasets are complicated and we want to make a simplification on them.

3.1.1 Data Processing for Project_id Classification

For the data processing of the classification problem on 'project_id'. Firstly, we work on the dataset 'interest_group_info' and create a new file called 'interest_group_info_a'. As shown in the graph 3.1a, The steps are as follows:

1. Delete the column 'description' which is useless for the classification since we focuses on the project id classification.

2. The 'interest_id' is ranked in an ascending order and there are totally 42 different 'interest_id'.

Secondly, considering the dataset 'user_mapping', we make some modifications and the new file is called 'user_mapping_a' as shown in the graph 3.1b. These modifications steps are as following:

1. Delete the rows where 'standup_id' has no 'slack_id', which should be seen as missing data and useless for the classification process.

2. Rank the 'standup_id' in an ascending order to make it convenient for data matching from different files.

	A	B	C
1		interest_id	name
2	1	2	_comms_technology
3	2	3	_comms_security
4	3	4	_comms_information
5	4	5	_comms_knowledge
6	5	6	_comms_infrastructure
7	6	7	_humans_wellbeing
8	7	8	_humans_recruit
9	8	9	_humans_retain
10	9	12	_humans_beta
11	10	13	_humans_payroll
12	11	14	_humans_benefits
13	12	15	_humans_gatherings
14	13	16	_hype_events
15	14	17	_hype_materials
16	15	18	_hype_socialmedia
17	16	19	_hype_speaking
18	17	20	_hype_campaigns
19	18	21	_hype_articles
20	19	22	_hype_branding
21	20	23	_leads_prospect
22	21	24	_leads_proposals
23	22	25	_connect_media
24	23	26	_connect_academia
25	24	27	_connect_partners
26	25	28	_connect_researchers
27	26	29	_x_navigation
28	27	30	_x_frontend
29	28	31	_x_softwareengineering
30	29	32	_x_optimisation

interest_group_info_a

(a) interest_group_info_a

	A	B
1	standup_id	slack_id
2	3	U02EENFGK
3	4	U02EBRG16
4	5	U0LNDAUQ6
5	6	U06U2FKJ4
6	9	U02EEHT1J
7	10	U1K1P0ENR
8	11	U054SL2DJ
9	12	U0ZU8KFAT
10	14	U0LN2N1M1
11	16	U0C9GGXLH
12	17	U0HJTJATX
13	23	U04PVV50R
14	26	U1G84H2MN
15	27	U0LMZS9LL
16	30	U0JDV9JG2
17	32	U0LUANYAH
18	33	U02EEJB7E
19	42	U1S309HPG
20	47	U1SHH9F5M
21	50	U0ZEL9V0F
22	51	U1MMCC6LU
23	52	U056HCTHB
24	53	U0C7MPQNP
25	55	U033RN09L
26	56	U1Q840YMT
27	58	U0CT2TPH8
28	59	U1JQT3G6B
29	62	U14K5KWC8
30	63	U1STRDDQB

user_mapping_a

(b) user_mapping_a

Figure 3.1: Table Preprocessing

Furthermore, for features of 'user_id' and 'project_id', we create two new files 'user_id_mapping' and 'project_id_mapping' to make a new map with the orders from small to large (Graph3.2a, Graph3.2b).There are totally 58 different 'user_id' and 83 different 'project_id' which is completely matchable, useful and meaningful.

Thirdly, we make the decision to not use 'Channel_info' dataset in this project because there are a lot of missing data information in it, which is just meaningless for the data mapping process.

For the fourth dataset: 'all_standups',we address it as following:

	A	B	C
1	user_id	number_map	
2	3	1	
3	4	2	
4	5	3	
5	6	4	
6	9	5	
7	10	6	
8	11	7	
9	12	8	
10	14	9	
11	16	10	
12	17	11	
13	23	12	
14	26	13	
15	27	14	
16	30	15	
17	32	16	
18	33	17	
19	42	18	
20	47	19	
21	50	20	
22	51	21	
23	53	22	
24	55	23	
25	56	24	
26	58	25	
27	59	26	
28	62	27	
29	63	28	
30	65	29	

user_id_mapping

(a) User_id_mapping

	A	B	C
1	project_id	mapping_number	
2	1	1	
3	2	2	
4	4	3	
5	7	4	
6	16	5	
7	17	6	
8	24	7	
9	31	8	
10	32	9	
11	33	10	
12	36	11	
13	37	12	
14	39	13	
15	40	14	
16	41	15	
17	44	16	
18	50	17	
19	61	18	
20	67	19	
21	70	20	
22	71	21	
23	72	22	
24	73	23	
25	74	24	
26	75	25	
27	76	26	
28	78	27	
29	83	28	
30	100	29	

project_id_mapping

(b) Project_id-mapping

Figure 3.2: ID Mapping Table

1. Delete the first column ‘comment’ since it is just a description with sentences which is meaningless and cannot be used in this project.

2. The column ‘interest_name’ is deleted and the ‘interest_id’ column is kept.

3. Rank the ‘standup_date’ column to start from the date of ‘2017/6/21’ to ‘2018/6/21’.

4. Remove the time part of the ‘date’ because this project is operated on a daily basis.

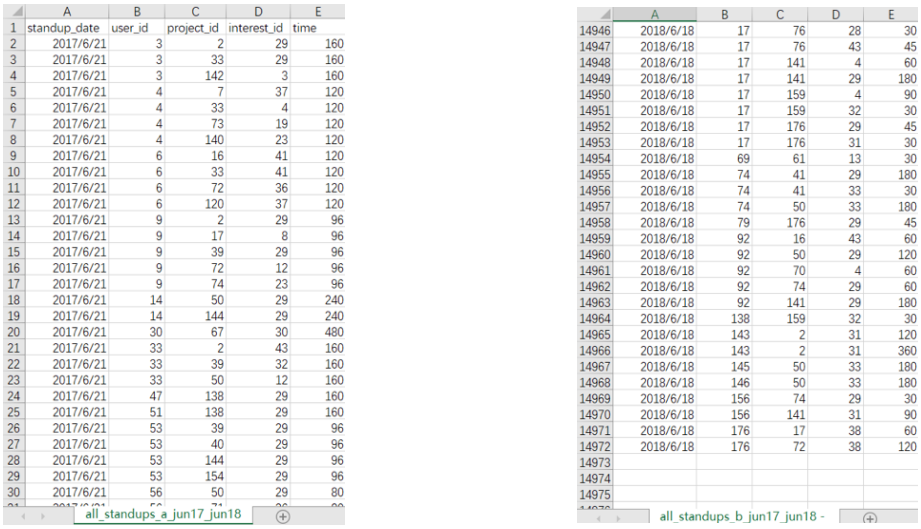
5. Delete the first two rows where some data information is missing.

6. Transform the column ‘time’ which is presented in a combination of hours and minutes format into the format of pure minutes.

7. Move the ‘standup_date’ to the first column.

8. Put the column ‘user_id’, ‘project_id’, ‘interest_id’ and ‘time’ from second to the 5-th column with an increasing rank.

The new ‘all_standups_a’ is shown in graph 3.3a.



	A	B	C	D	E
1	standup_date	user_id	project_id	interest_id	time
2	2017/6/21	3	2	29	160
3	2017/6/21	3	33	29	160
4	2017/6/21	3	142	3	160
5	2017/6/21	4	7	37	120
6	2017/6/21	4	33	4	120
7	2017/6/21	4	73	19	120
8	2017/6/21	4	140	23	120
9	2017/6/21	6	16	41	120
10	2017/6/21	6	33	41	120
11	2017/6/21	6	72	36	120
12	2017/6/21	6	120	37	120
13	2017/6/21	9	2	29	96
14	2017/6/21	9	17	8	96
15	2017/6/21	9	39	29	96
16	2017/6/21	9	72	12	96
17	2017/6/21	9	74	23	96
18	2017/6/21	14	50	29	240
19	2017/6/21	14	144	29	240
20	2017/6/21	30	67	30	480
21	2017/6/21	33	2	43	160
22	2017/6/21	33	39	32	160
23	2017/6/21	33	50	12	160
24	2017/6/21	47	138	29	160
25	2017/6/21	51	138	29	160
26	2017/6/21	53	39	29	96
27	2017/6/21	53	40	29	96
28	2017/6/21	53	144	29	96
29	2017/6/21	53	154	29	96
30	2017/6/21	56	50	29	80

all_standups_a_jun17_jun18

(a) All_stands_up_a

	A	B	C	D	E
14946	2018/6/18	17	76	28	30
14947	2018/6/18	17	76	43	45
14948	2018/6/18	17	141	4	60
14949	2018/6/18	17	141	29	180
14950	2018/6/18	17	159	4	90
14951	2018/6/18	17	159	32	30
14952	2018/6/18	17	176	29	45
14953	2018/6/18	17	176	31	30
14954	2018/6/18	69	61	13	30
14955	2018/6/18	74	41	29	180
14956	2018/6/18	74	41	33	30
14957	2018/6/18	74	50	33	180
14958	2018/6/18	79	176	29	45
14959	2018/6/18	92	16	43	60
14960	2018/6/18	92	50	29	120
14961	2018/6/18	92	70	4	60
14962	2018/6/18	92	74	29	60
14963	2018/6/18	92	141	29	180
14964	2018/6/18	138	159	32	30
14965	2018/6/18	143	2	31	120
14966	2018/6/18	143	2	31	360
14967	2018/6/18	145	50	33	180
14968	2018/6/18	146	50	33	180
14969	2018/6/18	156	74	29	30
14970	2018/6/18	156	141	31	90
14971	2018/6/18	176	17	38	60
14972	2018/6/18	176	72	38	120
14973					
14974					
14975					

all_standups_b_jun17_jun18 -

(b) All_stands_up_b

Figure 3.3: All_stands_up Table Processing

Concerning the fifth dataset ‘Slack_Private’, the time period between the dataset ‘all_stand_ups_a’ and the dataset ‘Slack_Private’ is different: the date in dataset ‘all_standups_a’ is from the date of ‘2017/6/21’ to ‘2018/6/21’ while the date in dataset ‘Slack_Private’ is from ‘2014/7/27’ to ‘2018/6/18’. In order to make the date be consistent and simplify the dataset, we take some steps as following:

1. Delete the data information in ‘Slack_Private’ originating at ‘2014/7/27’ to ‘2017/6/20’.

2. Delete the data information of ‘all_stand_ups_1’ in the time interval: from ‘2018/6/19’ to ‘2018/6/21’ and create a new file ‘all_stand_ups_b’ (graph 3.3b)

3. Transform the ‘user_id’ and ‘members’ into the form of number with the use of ‘user_mapping_a’ file.

4. Remove the data of 'user_id' and 'members' which cannot be replaced by number after transformation.

5. Delete the 'user_id' data which belongs to that row and keep only the 'user_id' numbers of other employees in the 'members' column.

6. Replace the 'slack_id' form into the number form of 'user_id' for the column 'mentions' and replace the blank with number '0' if no mentions are involved. Then, move this column to the left of the column 'messages'.

7. Remove the column 'reactions' since its content is not clear for classification.

8. Rank the column 'user_id' and 'members' from small to large.

Then, a new file 'Slack_Private_a' is created (graph 3.4). There is still a problem: how to address the column 'message' which is presented in the text format. We solve this problem with quantification method. That is, the messages number for each pair of the members in each day is computed. The column 'message' is replaced with a new name 'number_message'. A new file 'Slack_Private_b' is created (graph 3.5). After this, we make a data integration on the basis of file 'Slack_Private_b'. The steps are as follow:

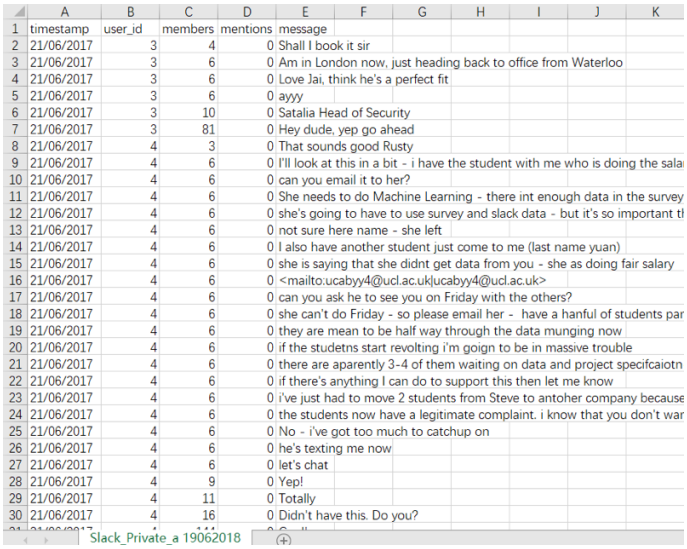
1. Make a summary for each 'user_id' in one cell for each date.

2. Make a summary of all the 'members' in one cell with a vector form.

3. Present the 'mentions' and the 'message_number' column in a vector format.

Then we create a new file 'Slack_Private_c'. (graph 3.6) We take an example to explain this file. Consider an employee whose 'user_id' is 16, in the data 21/06/2017, there is a conversation between this employee and other 3 employees with 'user_id' 4, 74 and 62 respectively with the related value of 'mentions' and 'message_number'. All the numbers in the vector 'message_number' are added up to create a new variable called 'messages' for the data training.

For the last dataset 'Slack_Public', it contains the public information of Satalia. We will not use it in this project since we want to use the 'Slack' information



	A	B	C	D	E
1	timestamp	user_id	members	mentions	message
2	21/06/2017	3	4	0	Shall I book it sir
3	21/06/2017	3	6	0	Am in London now, just heading back to office from Waterloo
4	21/06/2017	3	6	0	Love Jai, think he's a perfect fit
5	21/06/2017	3	6	0	ayyy
6	21/06/2017	3	10	0	Satalia Head of Security
7	21/06/2017	3	81	0	Hey dude, yep go ahead
8	21/06/2017	4	3	0	That sounds good Rusty
9	21/06/2017	4	6	0	I'll look at this in a bit - i have the student with me who is doing the sala
10	21/06/2017	4	6	0	can you email it to her?
11	21/06/2017	4	6	0	She needs to do Machine Learning - there int enough data in the survey
12	21/06/2017	4	6	0	she's going to have to use survey and slack data - but it's so important th
13	21/06/2017	4	6	0	not sure here name - she left
14	21/06/2017	4	6	0	I also have another student just come to me (last name yuan)
15	21/06/2017	4	6	0	she is saying that she didnt get data from you - she as doing fair salary
16	21/06/2017	4	6	0	<mailto:ucabyy4@ucl.ac.uk|ucabyy4@ucl.ac.uk>
17	21/06/2017	4	6	0	can you ask he to see you on Friday with the others?
18	21/06/2017	4	6	0	she can't do Friday - so please email her - have a hanful of students par
19	21/06/2017	4	6	0	they are mean to be half way through the data munging now
20	21/06/2017	4	6	0	if the studetns start revolting i'm goign to be in massive trouble
21	21/06/2017	4	6	0	there are aparently 3-4 of them waiting on data and project specifcaiotn
22	21/06/2017	4	6	0	if there's anything I can do to support this then let me know
23	21/06/2017	4	6	0	i've just had to move 2 students from Steve to antoher company because
24	21/06/2017	4	6	0	the students now have a legitimate complaint. i know that you don't war
25	21/06/2017	4	6	0	No - i've got too much to catchup on
26	21/06/2017	4	6	0	he's texting me now
27	21/06/2017	4	6	0	let's chat
28	21/06/2017	4	9	0	Yep!
29	21/06/2017	4	11	0	Totally
30	21/06/2017	4	16	0	Didn't have this. Do you?

Slack_Private_a 19062018

Figure 3.4: Slack_Private_a

to make a prediction on the 'project_id'. Nevertheless, this public information involves nearly every issue about the company and only a small amount of data are related with the project worked by employees, which is not appropriate to be used for classification.

However, at present, we still have not linked the 'stand_up' data with the 'slack_data'. A new file should be created to use the 'slack_data' to make a classification on 'stand_up' data. We do this as follows:

1. Make a summary of the mappings between 'user_id' and 'project_id' of the dataset 'All_stand_ups_b' and then create the file 'label_file' (figure 3.7a).

2. Make a link between the 'slack' information and the 'stand_ups' data by considering the file 'Slack_Private_c' and 'label_file'. Replace the 'user_id' which is not in the file 'Slack_Private_Final' with mark '?' in the file 'label_file'. The new file is 'label_file_1' (figure 3.7b).

3. Replace the 'user_id' which is not in the file 'final_label' with mark '?' in the file 'Slack_Private_c', the new file is 'Standup_final_1' (figure 3.8a).

4. Delete all the rows with mark '?' for both file 'label_file_1' and 'Standup_final_1'.

5. Create a new file 'Standup_final_2' that links the file 'label_file_1' and 'Standup_final_1'. It is presented in figure 3.8b,

	A	B	C	D	E
1	timestamp	user_id	members	mentions	number_message
2	21/06/2017	3	4	0	1
3	21/06/2017	3	6	0	3
4	21/06/2017	3	10	0	1
5	21/06/2017	3	81	0	1
6	21/06/2017	4	3	0	1
7	21/06/2017	4	6	0	19
8	21/06/2017	4	9	0	1
9	21/06/2017	4	11	0	1
10	21/06/2017	4	16	0	1
11	21/06/2017	4	144	0	1
12	21/06/2017	5	23	0	13
13	21/06/2017	5	23 26	0	1
14	21/06/2017	6	3	6	9
15	21/06/2017	6	4	6	24
16	21/06/2017	6	16	0	1
17	21/06/2017	6	50	6	10
18	21/06/2017	6	56	0	3
19	21/06/2017	6	69	0	6
20	21/06/2017	6	83	0	5
21	21/06/2017	6	85	0	8
22	21/06/2017	6	152	0	1
23	21/06/2017	6	14 56	0	2
24	21/06/2017	6	16 138	0	1
25	21/06/2017	6	16 14	0	2
26	21/06/2017	6	16 79	0	4
27	21/06/2017	6	4 17	0	2
28	21/06/2017	6	53 16 56	0	4
29	21/06/2017	6	74 79	0	6
30	21/06/2017	6	83 141	0	9

Slack_Private_b

Figure 3.5: Slack_Private_b

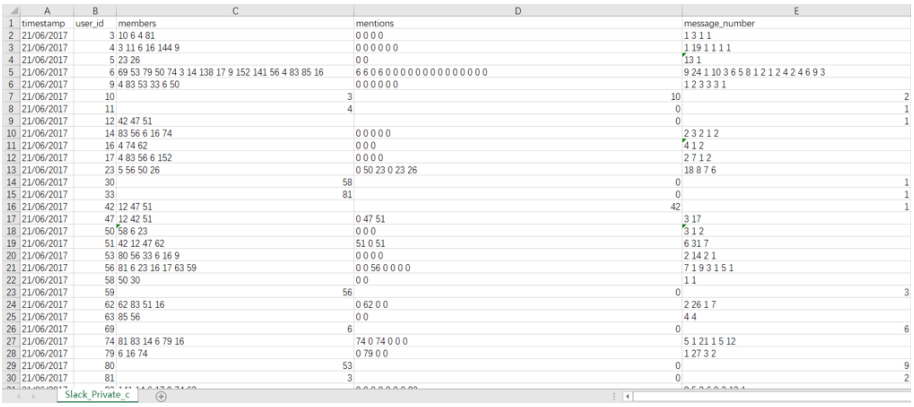
Furthermore, we denote the ‘user_id’, ‘members’, ‘mentions’ and ‘project_id’, ‘project_id’ in a vector form with ‘one-hot encoding’. ‘user_id’, ‘members’ and ‘mentions’ are the form of a 58×1 vector. ‘project_id’ is in the format of an 83×1 vector. There are totally 5027 rows of the data information in the final ‘Standup_final_2’.

3.1.2 Data Processing for Prediction on Time

Concerning the prediction on ‘time’, since we focus on making a prediction on the ‘time’ by using the employee’s ‘user_id’,’project_id’ and ‘interest_id’. We just use the file ‘all_standups_a’, which has been processed in the data processing part for ‘project_id’ classification. There are totally 15005 rows of the data information.

3.2 Methodology

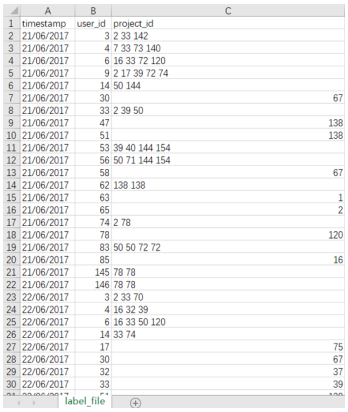
In this part,we make a detailed description on the algorithms used in this project.



	A	B	C	D	E
1	timestamp	user_id	members	mentions	message_number
2	21/06/2017	3	10 6 4 81	0 0 0 0	1 3 1 1
3	21/06/2017	4	3 11 6 16 144 9	0 0 0 0 0 0	1 19 1 1 1 1
4	21/06/2017	5	23 26	0 0	13 1
5	21/06/2017	6	69 53 79 50 74 3 14 138 17 9 152 141 56 4 83 85 16	6 6 0 6 0 0 0 0 0 0 0 0 0 0 0 0 0 0	9 24 1 10 3 6 5 8 1 2 1 2 4 2 4 6 9 3
6	21/06/2017	9	4 83 53 33 6 50	0 0 0 0 0 0	1 2 3 3 3 1
7	21/06/2017	10	3	10	2
8	21/06/2017	11	4	0	1
9	21/06/2017	12	42 47 51	0	1
10	21/06/2017	14	83 56 6 16 74	0 0 0 0 0	2 3 2 1 2
11	21/06/2017	16	4 74 62	0 0 0	4 1 2
12	21/06/2017	17	4 83 56 6 152	0 0 0 0	2 7 1 2
13	21/06/2017	23	5 56 50 26	0 50 23 0 23 26	18 8 7 6
14	21/06/2017	30	58	0	1
15	21/06/2017	33	81	0	1
16	21/06/2017	42	12 47 51	42	1
17	21/06/2017	47	12 42 51	0 47 51	3 17
18	21/06/2017	50	58 6 23	0 0 0	3 1 2
19	21/06/2017	51	42 12 47 62	51 0 51	6 31 7
20	21/06/2017	53	80 56 33 6 16 9	0 0 0 0	2 14 2 1
21	21/06/2017	56	81 6 23 16 17 63 59	0 0 56 0 0 0	7 1 9 3 1 5 1
22	21/06/2017	58	50 30	0 0	1 1
23	21/06/2017	59	56	0	3
24	21/06/2017	62	62 83 51 16	0 62 0 0	2 26 1 7
25	21/06/2017	63	85 56	0 0	4 4
26	21/06/2017	69	6	0	6
27	21/06/2017	74	81 83 14 6 79 16	74 0 74 0 0 0	5 1 21 1 5 12
28	21/06/2017	79	6 16 74	0 79 0 0	1 27 3 2
29	21/06/2017	80	53	0	9
30	21/06/2017	81	3	0	2

Slack_Private_c

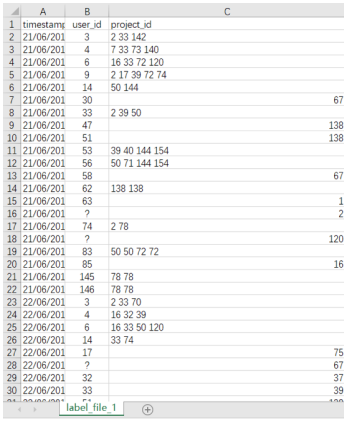
Figure 3.6: Final Result Slack_Private_c



	A	B	C
1	timestamp	user_id	project_id
2	21/06/2017	3	2 33 142
3	21/06/2017	4	7 33 73 140
4	21/06/2017	6	16 33 72 120
5	21/06/2017	9	2 17 39 72 74
6	21/06/2017	14	50 144
7	21/06/2017	30	67
8	21/06/2017	33	2 39 50
9	21/06/2017	47	138
10	21/06/2017	51	138
11	21/06/2017	53	39 40 144 154
12	21/06/2017	56	50 71 144 154
13	21/06/2017	58	67
14	21/06/2017	62	138 138
15	21/06/2017	63	1
16	21/06/2017	65	2
17	21/06/2017	74	2 78
18	21/06/2017	78	120
19	21/06/2017	83	50 50 72 72
20	21/06/2017	85	16
21	21/06/2017	145	78 78
22	21/06/2017	146	78 78
23	22/06/2017	3	2 33 70
24	22/06/2017	4	16 32 39
25	22/06/2017	6	16 33 50 120
26	22/06/2017	14	33 74
27	22/06/2017	17	75
28	22/06/2017	30	67
29	22/06/2017	32	37
30	22/06/2017	33	39

label_file

(a) Label_file



	A	B	C
1	timestamp	user_id	project_id
2	21/06/201	3	2 33 142
3	21/06/201	4	7 33 73 140
4	21/06/201	6	16 33 72 120
5	21/06/201	9	2 17 39 72 74
6	21/06/201	14	50 144
7	21/06/201	30	67
8	21/06/201	33	2 39 50
9	21/06/201	47	138
10	21/06/201	51	138
11	21/06/201	53	39 40 144 154
12	21/06/201	56	50 71 144 154
13	21/06/201	58	67
14	21/06/201	62	138 138
15	21/06/201	63	1
16	21/06/201	?	2
17	21/06/201	74	2 78
18	21/06/201	?	120
19	21/06/201	83	50 50 72 72
20	21/06/201	85	16
21	21/06/201	145	78 78
22	21/06/201	146	78 78
23	22/06/201	3	2 33 70
24	22/06/201	4	16 32 39
25	22/06/201	6	16 33 50 120
26	22/06/201	14	33 74
27	22/06/201	17	75
28	22/06/201	?	67
29	22/06/201	32	37
30	22/06/201	33	39

label_file_1

(b) Modified Label_file$_1$

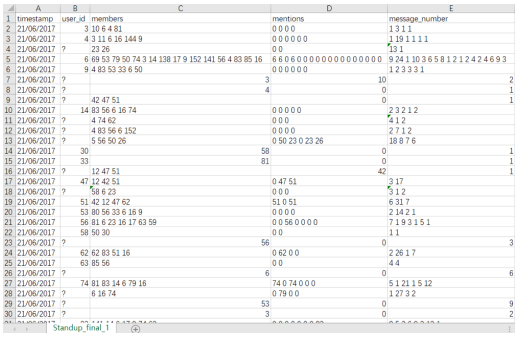
Figure 3.7: Label File Processing

3.2.1 Classification for Stand_up Data

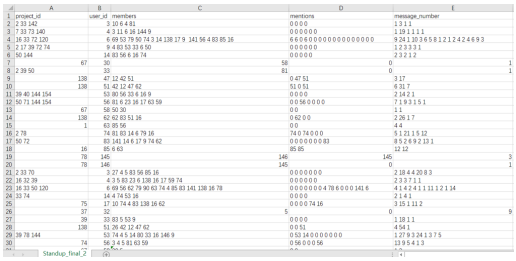
For the 'stand up' data classification, we use multiple neural network and Gaussian Naive Bayesian algorithm. More specifically, as shown in graph3.9, we make a classification for the output 'project_id' with the 'user_id', 'members', 'mentions' and 'message_number' as the inputs.

3.2.1.1 Multiple Neural Network

The Multiple Neural network is an algorithm which is similar to a human brain. By recognizing the patterns, it makes a computer learn from the observed data. More specifically, it obtains the knowledges with a learning procedure [25]. Then, the acquired knowledges are stored in synaptic weights. The network synaptic weights



(a) Standup_final_1



(b) Standup_final_2

Figure 3.8: Stand_up_file Processing

are changed for achieving the desired objective during the learning process. Similar to the human brain, the Multiple Neural networks operate like a non-linear parallel information-processing system that perform some computations rapidly including pattern perception and recognition [20]. The clustering or labeling raw inputs are used to interpret the sensory data. The images, sound, text are translated into a vector form and are presented in the recognized patterns.

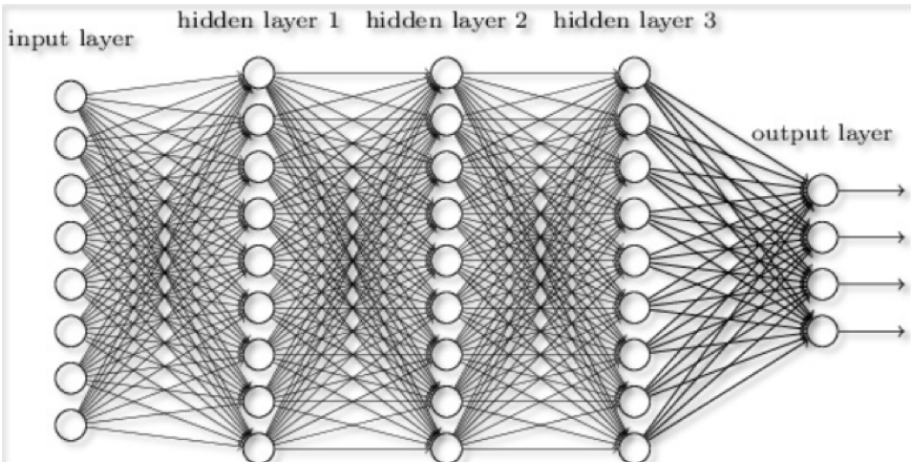


Figure 3.9: Multiple Neural network

3.2.1.1.1 Neural Networks Structure As shown in the graph3.9, there are three different layers in the Multiple Neural network system including [58]:

- Input layer,
- Hidden layer,

- Output layer.

For each layer, there are one or more nodes, which are presented in small circles. The lines between the different nodes represent the information flow from one node to the next. The information flows from the left to the right. In other words, the information flows from the input layer to the output layer [73]. The input layer's nodes are passive, which means they do not change the data. A single value is received on the input and then the value is duplicated to their multiple outputs. On the contrary, the nodes of both the hidden and output layer are active, which means they can change the data [73]. The Multiple Neural network system is an interconnected structure where the values from the input layer are copied and then sent to all the hidden nodes [2]. When the values entering a hidden node, they are multiplied by weights which are the determined numbers stored in the computer program. After this, a single number is produced by adding the weighted inputs. Generally, Multiple Neural network system can have layers with any number and any number of nodes per layer. There are basically three different parameters in the system. The first one is the number of hidden layers (L). The second one is the number of nodes per layer (N) and the third one is the learning rate (R) [54].

Mathematically, we can present the components of the neural network system as following:
The Neural Network system has the components including Neurons,Connections, Weights, Biases and a Propagation Funtion

Neurons

A neuron which receives an input $p_j(t)$ from predecessor neurons with label j includes some components [74]:

- an activation $a_j(t)$ which depends on a discrete parameter of time.
- a threshold θ_j, which remains fixed unless it is altered with a learning function.
- an activation function f which makes a computation of the new activation at

a given time t+1 with $a_j(t)$, θ_j and the input $p_j(t)$:

$$a_j(t+1) = f(a_j(t), \theta_j, p_j(t)) \tag{3.2.1}$$

- an output function f_{out} which computes the output by the activation:

$$o_j(t) = f_{out}(a_j(t)) \tag{3.2.2}$$

Connections, Weights & Biases

The neural network system is composed of connections which transfer the neuron i's output to the input of the neuron j. In that case, j is the successor of i and i is the predecessor of j. A weight w_{ij} [74]. is assigned to each connection. In some cases, total weighted sum of inputs is added with a bias term which serves as the threshold to change the activation function [75].

Propagation Function The propagation function makes a computation on the input $p_j(t)$ for the neuron j with the outputs $a_i(t)$ of predecessor neurons [74]

$$p_j(t) = \sum_i o_i(t) w_{ij} \tag{3.2.3}$$

When the function is added with a bias value, the function changes to [12]:

$$p_j(t) = \sum_i o_i(t) w_{ij} + w_{0j} \tag{3.2.4}$$

where w_{0j} is a bias.

We select Multiple Neural Network algorithm because of its advantages as mentioned in the literature review: not many restrictions on the input variables in classification; can make a generalization after the learning process. In this project, we set the number of hidden layers L to be 2 and set the learning rate R at 0.1. We change the number of nodes per layer N to make a comparison of different results. Then, we following the process presented above to running the Neural Network model.

3.2.2 Gaussian Naive Bayesian

For the Gaussian Naive Bayesian algorithm, it is a specific case of the general Naive Bayes model which is shown below.

3.2.2.1 Probabilistic Model

Naive Bayes is a conditional probability model. When a problem instance with n features (a vector $X = (X_1, ..., X_n)$)is to be classified. Naive Bayes assigns to this instance with probabilities $p(C_k|x_1, ..., x_n)$ [49].By using Bayes' theorem, the conditional probability can be written as:

$$p(C_k|x) = \frac{p(C_k)p(x|C_k)}{p(x)} \tag{3.2.5}$$

Also, with the use of Bayesian probability terminology, the above equation can be written as:

$$posterior = \frac{prior * likelihood}{evidence} \tag{3.2.6}$$

In practice, the denominator is independent on C with the features x_i values being given. Only the numerator of that fraction is considered. Then, the denominator is substantially constant. With the use of the chain rule of the conditional probability, the numerator is equivalent to the joint probability model which is presented as follows :

$$\begin{aligned} p(C_k, x_1, ...x_n) &= p(x_1, ...x_n, C_k) \\ &= p(x_1|x_2, ...x_n, C_k)p(x_2|x_3, ..., x_n, C_k)p(x_3, ..., x_n, C_k) \\ &= ... \\ &= p(x_1|x_2, ...x_n, C_k)p(x_2|x_3, ..., x_n, C_k)...p(x_n - 1|x_n, C_k)p(x_n|C_k)p(C_k) \end{aligned} \tag{3.2.7}$$

There is a "naive" conditional independence assumption: given the category C_k, for $j \neq i$, each x_i feature is conditionally independent of every other feature x_j. That is:

$$p(x_i|x_{i+1}, ..., x_n, C_k) = p(x_i|C_k) \tag{3.2.8}$$

Then the joint model is as the equation3.2.9 showing:

$$\begin{aligned} p(C_k|x_1, ...x_n) \propto p(x_1, ...x_n, C_k) = \\ &= p(C_k)p(x_1|C_k)p(x_2|C_k)p(x_3|C_k)... \\ &= p(C_k)\prod_{i=1}^{n} p(x_i|C_k) \end{aligned} \tag{3.2.9}$$

Where $\propto$ represents proportionality. With the independence assumptions, the conditional distribution with the class variable C is in below equation3.2.10:

$$p(C_k|x_1, ..., x_n) = \frac{1}{Z} p(C_k) \prod_{i=1}^{n} p(x_i|C_k) \tag{3.2.10}$$

where $Z = p(x) = \sum_k p(C_k)p(x|C_k)$ is scaling factor depending only on $x_1, ..., x_n$.

3.2.2.2 Constructing a Classifier for the Probability Model

The naive Bayes classifier combines the naive Bayes probability model with a decision rule. The maximum a posteriori (MAP) decision rule is used as assumption. By assigning a class label $y = C_k$ for some k, the Bayes classifier is the function which can be written as in figure3.2.11[32]:

$$\hat{y} = argmax_{k \in 1,...,K} p(C_k) \prod_{i=1}^{n} p(x_i|C_k) \tag{3.2.11}$$

3.2.2.3 Parameter Estimation and Event Models

A class's prior is computed by an assumption on equiprobable classes (priors = 1 / number of classes) or by a class probability estimate from the training set (prior for a given class = number of samples in the class / total number of samples) [44].

3.2.2.4 Gaussian Naive Bayes

Then, we considering the naive Bayes in the Gaussian case.

When the continuous values associated with each class are distributed according to a Gaussian distribution, the model becomes Gaussian naive Bayes [45]. Suppose a continuous variable x is in the training data, the data are segmented by the class. Then, the mean and variance of x are computed in each class with μ_k denoting the mean in x for class C_k, and σ_k^2 being the corresponding variance. Suppose we have collected some observation value v. Then, given a class C_k, the probability distribution of v $p(x = v|C_k)$ with parameters by μ_k and $\frac{2}{k}$ can be computed as 3.2.12 [33]:

$$p(x = v|C_k) = \frac{1}{\sqrt{2\pi\sigma_k^2}} e^{-\frac{(v-\mu_k)^2}{2\sigma_k^2}} \tag{3.2.12}$$

The graph 3.10 below shows a decision boundary computed for a dataset with the use of Gaussian naive Bayes classification. The line represents the decision boundary corresponding to the curve where a new point can be part of each class with equal posterior probability. In that case, a classification with perfect contamination and completeness could be found [16].

In this project, we use Gaussian navie Bayes for multi-label classification since it has been proved to be more effective than some other approaches in terms of multi-label classification as mentioned in literature review.

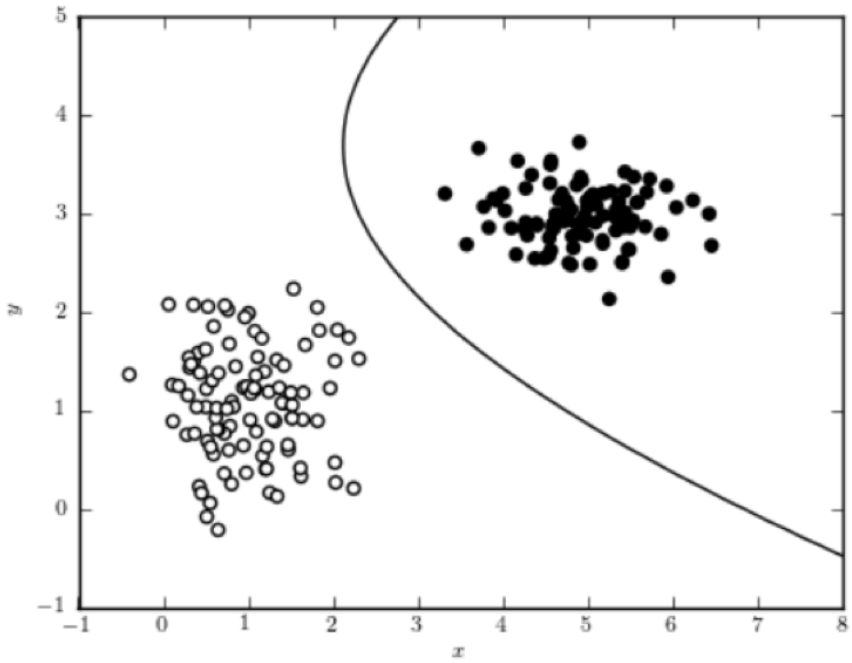


Figure 3.10: Gaussian Naive Bayes Classification

3.2.3 Regression on 'time'

3.2.3.1 Polynomial Regression Model

In general, when the expected value of y is predicted by x in a form of n-th degree polynomial, the polynomial regression model is [43]

$$y = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 + ... + \beta_n x^n + \epsilon \tag{3.2.13}$$

where ϵ is a random error with a zero mean under the condition of a scalar variable x.

In matrix form. the polynomial regression model:

$$y_i = \beta_0 + \beta_1 x_i + \beta_2 x_i^2 + \beta_3 x_i^3 + ... + \beta_m x_i^m + \epsilon_i (i = 1, 2, ..., n) \tag{3.2.14}$$

can be presented with a response vector y, a matrix X, a parameter vector β, and a random errors vector ϵ. The i-th row of X and y contains the value of x and y for the i-th data sample. The model is presented a linear equations system [7]: when using matrix notation, it can be written as:

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} 1 & x_1 & x_1^2 & \dots & x_1^m \\ 1 & x_2 & x_2^2 & \dots & x_2^m \\ 1 & x_3 & x_3^2 & \dots & x_3^m \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_n & x_n^2 & \dots & x_n^m \end{bmatrix} \begin{bmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ \vdots \\ \beta_m \end{bmatrix} + \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \varepsilon_3 \\ \vdots \\ \varepsilon_n \end{bmatrix},$$

Figure 3.11: Polynomial Regression Model in Matrix

$$y = X\beta + \epsilon \tag{3.2.15}$$

For parameter estimation, the model is linear in terms of the parameters $\beta_0, \beta_1, .., \beta_m$. the computational and inferential process of polynomial regression can be addressed with multiple regression, which is solved by treating $x, x^2, ..., x^m$ as the distinct independent variables in a multiple regression model [59]. Polynomial regression model is usually fit through the least squares method which minimizes the variance of the unbiased estimators of the coefficients, under the Gauss–Markov theorem [18]. More specifically, in order to estimate the unknown parameters $\beta_0, \beta_1, .., \beta_m$. We use the Maximum Likelihood Estimation (MLE) method.

The vector of estimated polynomial regression coefficients (using the Maximum Likelihood estimation) is 3.2.16:

$$\hat{\overrightarrow{\beta}} = (X^T X)^{-1} X^T \overrightarrow{y} \tag{3.2.16}$$

There is a requirement that the matrix should be invertible, with an assumption m < n in equation (3.2.16). X is a Vandermonde matrix and all the x_i values are distinct under the invertibility condition. For the parameter estimator result β. It is the unique least-squares solution [21].

We select the polynomial regression algorithm because as discussed in literature review, it can fit a wider range of functions compared with many other machine learning algorithms. In the prediction process of this project, the parameter β is estimated first. Then, we fit the model into the data to obtain a result.

3.2.3.2 Random Forest Regression

For the random forest regression algorithm, it is a development from the decision tree algorithm. A decision tree is a tree whose nodes represent random transitions (a circular), decisions (a square) or terminal nodes [36]. The branches or edges are binary (true/false, yes/no) which denote the possible paths between one node and the other. When using a decision tree for regression or classification, given a row of data or a set of features, the process begins from the root and then passes through all the subsequent decision nodes to the terminal node. There is an example to illustrate this, which is shown in graph 3.12, when a person wants to buy a new car, there is a dataset of different cars which have three features: Displacement (Numeric), Car Drive Type (Categorical), and Clearance (Numeric). The decision tree algorithm can be used as shown in figure3.12 [41]: The tree root is the decision node to split the dataset by using a feature or variable. The best splitting metric evaluated for each class in the dataset is obtained. Following the splitting metric at each decision node, the decision tree learns through recursively splitting the dataset from the root in a node by node mode. When the splitting metric is arriving at a global extremum, the terminal nodes are achieved.

Bootstrap Aggregation Bootstrap aggregation is a powerful technique which can improve the learning outcome on datasets with a decrease in model variances (overfitting). In general, bootstrap aggregation with ensemble models is an effective method to reduce the variance and solve the overfitting problem of learning models by using bootstrap samples and aggregating the learning models' outputs

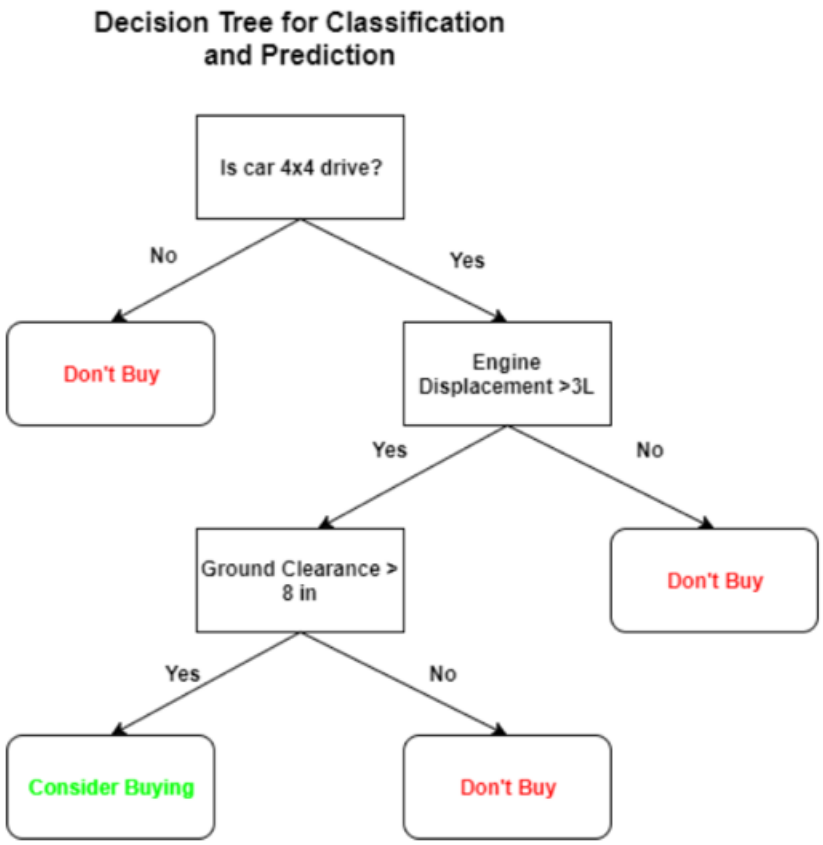


Figure 3.12: Gaussian Naive Bayes Classification

[42]. Bootstrap aggregation can be applied to any supervised learning algorithms. It works by creating M subsets with n samples per subset from the original dataset where the n samples are uniformly sampled with replacement [65]. The graph 3.13 below illustrates this: At the first step, for each data point, the labels are preserved. That is, the data tuples (X,Y) are sampled where Y is a vector of responses and X is a vector of inputs. In theory, bootstrap aggregation converges to the mean of the non-bagged function estimator when all the samples from the original dataset are used as the number of bootstrap samples M approaches infinity. In the stochastic gradient learning, as the gradient estimate has the tendency to be 'jostled around' more when overcoming local extrema, learning in random order with repetitions from multiple data sample tends to increase learning performance. Bootstrap aggregation has the tendency to add bias to the bagged estimator to reduce variance. Meanwhile, the increase in bias is small compared to

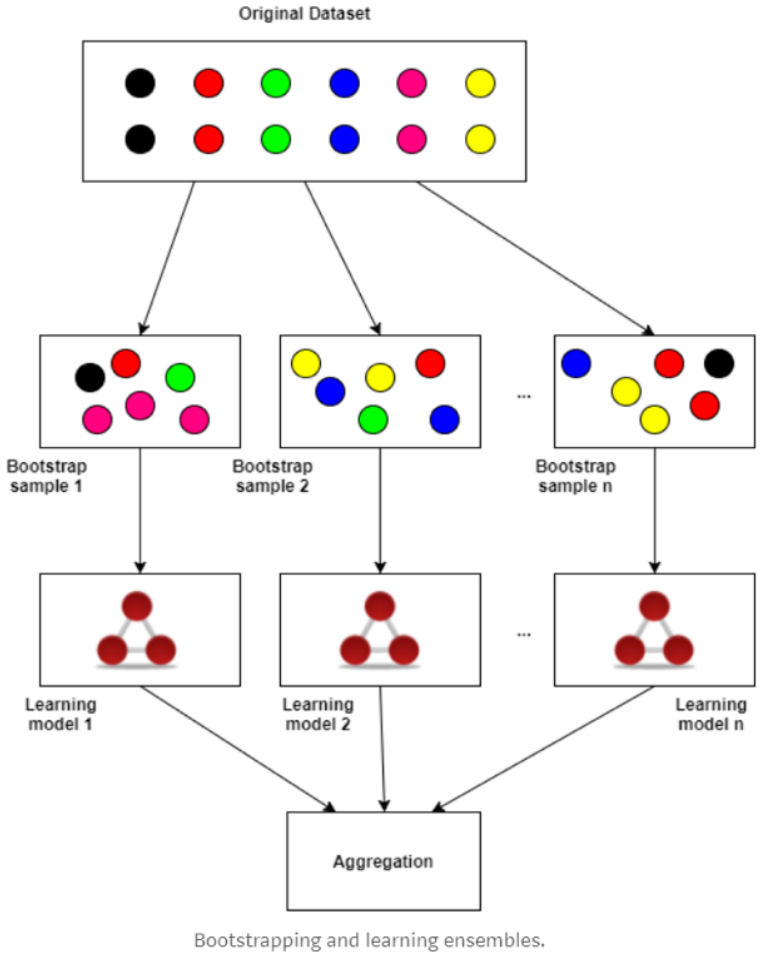


Figure 3.13: Bootstrapping and Learning Ensembles

the variance reduction. At the second step, for each M bootstrap sample, k individual learning models are created. The individual learning model outputs are then aggregated or averaged [65].

Random forest algorithm is a combination of Decision Trees and Bootstrap aggregation[29]. It consists of an ensemble of k untrained decision trees. For each tree, it has only a root node with M bootstrap samples trained by the feature bagging method or a variant of the random subspace method. The procedure of a random forest training is as below [23]:

1. Select p features randomly from available features D at the current node. Commonly, the total number of features D is much larger than the number of features p.

2. Compute the split point which is best for tree k by using the specified splitting metric (Information Gain, Gini Impurity, etc.) and then split the current node into subset nodes and reduce the features number D from the node.

3. Repeat steps 1 to 2 until either some extrema are reached by the splitting metric or a maximum tree depth i is achieved.

4. Repeat steps 1 to 3 in the forest for each tree k.

5. Aggregate on each tree's output in the forest.

Being different from single decision trees, at each split point, random forest algorithm splits by the way of selecting multiple feature variables rather than single features variables [29]. Intuitively, by using this feature bagging procedure, the variable selection properties of decision trees can be drastically improved.

We select random forest since as mentioned in literature review, it is convenient for interpolation and easy to see what variables or features are related to the regression and the relative importance on the basis of their location depth wise on the tree.

3.2.3.3 Support Vector Machine (SVM) Regression

The support vector machine (SVM) regression algorithm is a development from the traditional linear regression and non-linear regression. Graph 3.14 shows the traditional linear regression algorithm, given the training data $(x_1, y_1), ..., (x_n, y_n)$, y_i is the output vector and x_i is the input vector, the algorithm tries to find a combination of (w, b) in linear function $w^T x + b$, which is an optimal solution of:

$$min_{w,b} \sum_{i=1}^{n} (y_i - (w^T x_i + b))^2 \tag{3.2.17}$$

That is, $w^T x + b$ makes an approximation on the training data by minimizing the sum of square errors. Generally, the number of features F, is less than n. If it is not the case, a line passing through all points and the optimal function is $w^T x + b$, which results in over-fitting [24].

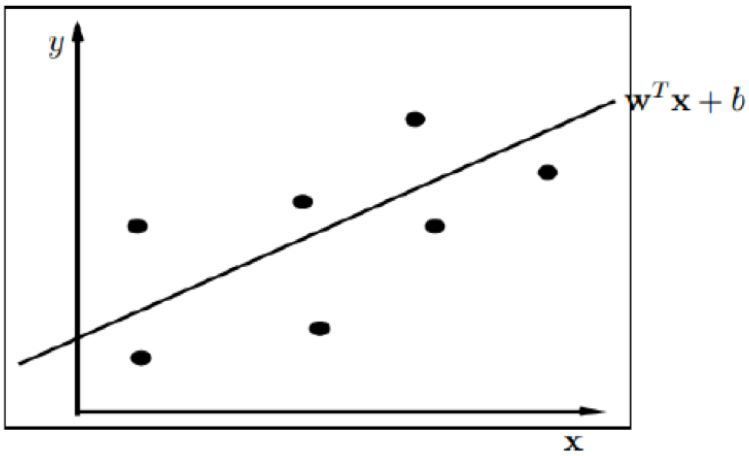


Figure 3.14: Bootstrapping and Learning Ensembles

An example of non-linear regression is shown in graph3.15. The data are mapped to a higher dimensional space with a function $\phi(x)$. When $F \leq$ dimensionality of $\phi(x)$, over-fitting happens [17].

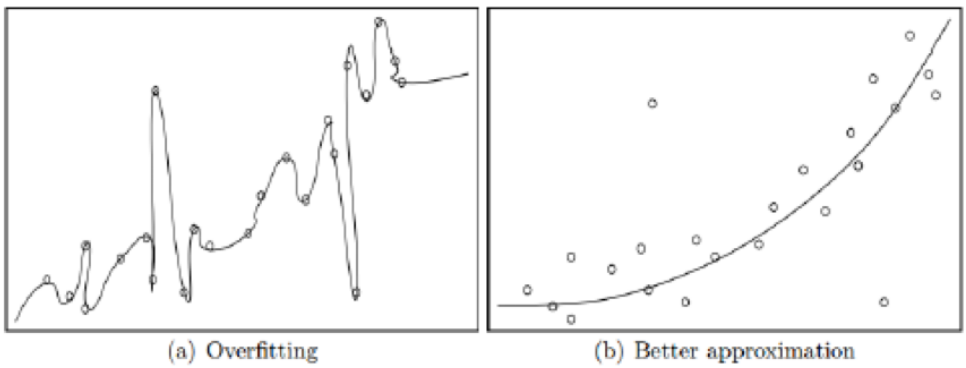


Figure 3.15: Bootstrapping and Learning Ensembles

SVM regression addresses the over-fitting problem after using ϕ with the following reformulation of 3.2.17 (figure 3.20) [64]: The equation 3.2.17 (with $\phi(x)$ replacing x) is equivalent to the equation showed in figure 3.16, if (w, b, ξ, ξ^*) is optimal for equation stated above, then $\xi_i^2 + (\xi_i^*)^2$ is minimized, we get:

$$\begin{aligned} \xi_i &= max(y_i - (w^T\phi(x_i) + b), 0) \\ \xi_i^* &= max(-y_i - (w^T\phi(x_i) + b), 0) \\ \xi_i^2 + (\xi_i^*)^2 &= (y_i - (w^T\phi(x_i) + b))^2 \end{aligned} \tag{3.2.18}$$

$$\begin{aligned} \min_{\mathbf{w},b,\boldsymbol{\xi},\boldsymbol{\xi}^*} \quad & \sum_{i=1}^{n} \xi_i^2 + (\xi_i^*)^2 \\ \text{subject to} \quad & -\xi_i^* \leq y_i - (\mathbf{w}^\mathsf{T}\phi(\mathbf{x}_i) + b) \leq +\xi_i \\ & \xi_i, \xi_i^* \geq 0 \; i = 1, \ldots, n \end{aligned}$$

Figure 3.16: Lagrange Equation for SVM

Also, at an optimal solution. $\xi_i \xi_i^* = 0$ When using linear errors, we have: To avoid

$$\begin{aligned} \min_{\mathbf{w},b,\boldsymbol{\xi},\boldsymbol{\xi}^*} \quad & \sum_{i=1}^{l} (\xi_i + \xi_i^*) \\ \text{subject to} \quad & -\xi_i^* \leq y_i - (\mathbf{w}^\mathsf{T}\phi(\mathbf{x}_i) + b) \leq +\xi_i \\ & \xi_i, \xi_i^* \geq 0 \; i = 1, \ldots, l \end{aligned}$$

Figure 3.17: Linear Errors Avoiding

overfitting, a threshold ϵ is given, SVM regression uses two modifications so that the i-th datum satisfies:

$$-\epsilon \leq y_i - (w^T\phi(x_i) + b) \leq \epsilon \tag{3.2.19}$$

Then $\xi_i = \xi_i^* = 0$ when it is a correct approximation. An additional term $w^T w$ is added to smooth the objective function $w^T\phi(x_i) + b$. Then, svm regression solves the following problem of optimization [67]:

$$\begin{aligned} \min_{w,b,\boldsymbol{\xi},\boldsymbol{\xi}^*} \quad & \frac{1}{2}\mathbf{w}^\mathsf{T}\mathbf{w} + C\sum_{i=1}^{l} (\xi_i + \xi_i^*) \\ \text{subject to} \quad & (\mathbf{w}^\mathsf{T}\phi(\mathbf{x}_i) + b) - y_i \leq \epsilon + \xi_i \\ & y_i - (\mathbf{w}^\mathsf{T}\phi(\mathbf{x}_i) + b) \leq \epsilon + \xi_i^* \\ & \xi_i, \xi_i^* \geq 0 \; i = 1, \ldots, l \end{aligned}$$

Figure 3.18: Final Lagrange Equations

Where ϵ_i and ϵ_i^* are the upper and the lower training error respectively. With the constraint of the ϵ-insensitive tube $|y_i - (w^T\phi(x_i) + b)| \leq \epsilon$. This can be seen from figure 3.20. The error ϵ_i or ϵ_i^* is expected minimised in the objective function

when x_i is not in the tube. Under-fitting and over-fitting of the training data can be avoided in SVM regression by minimizing the training error $C\sum_{i=1}^{l}(\xi_i+\xi_i^*)$ and the regularization term $\frac{1}{2}w^Tw$. From graph3.21, we would like the approximate function to be as general as possible to represent the data distribution when the training data are in the ϵ-insensitive tube. The width of the tube ϵ, The cost of error C, and the mapping function ϕ are the parameters controlling the regression quality. After solving the dual problem: Where $K_{ij} = k(x_i, x_j) = \phi(x_i)^T\phi(x_j)$. The

$$\begin{aligned} \min_{\alpha,\alpha^*} \quad & \frac{1}{2}(\boldsymbol{\alpha}-\boldsymbol{\alpha}^*)^{\mathsf{T}}\mathbf{K}(\boldsymbol{\alpha}-\boldsymbol{\alpha}^*)+\epsilon\sum_{i=1}^{n}(\alpha_i+\alpha_i^*)+\sum_{i=1}^{n}y_i(\alpha_i-\alpha_i^*) \\ \text{subject to} \quad & \sum_{i=1}^{n}(\alpha_i-\alpha_i^*)=0, 0\le a_i, a_i^*\le C,\ i=1,\ldots,n \end{aligned}$$

Figure 3.19: Dual Problems

primal-dual relation is:

$$w = \sum_{i=1}^{n}(-\alpha_i+\alpha_i^*)\phi(x_i) \tag{3.2.20}$$

The approximate function is:

$$\sum_{i=1}^{n}(-\alpha_i+\alpha_i^*)k(x_i,x_j)+b \tag{3.2.21}$$

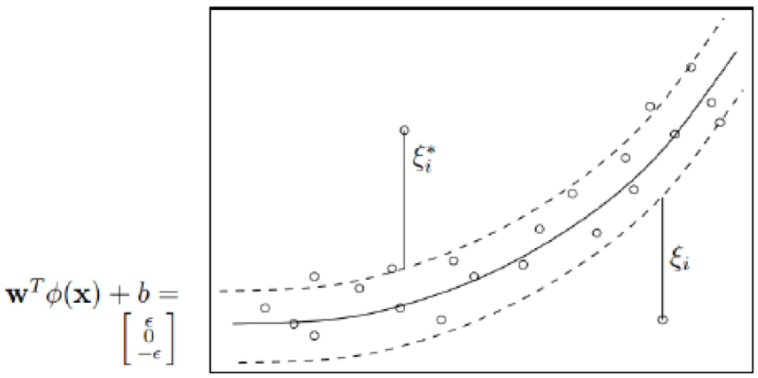


Figure 3.20: Support Vector Machine Regression

We select SVM in this project beacuase as noted in literature review, it has a parameter on regularization which avoids the problem of over-fitting with a lower error rate in testing process.

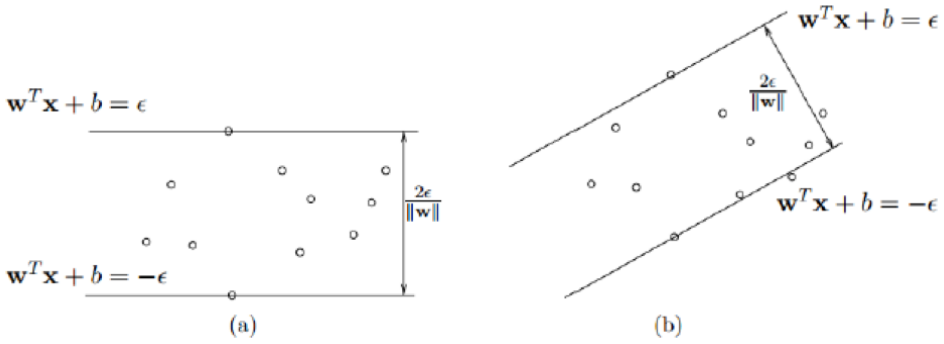


Figure 3.21: More General Approximate Function Through a Maximum of the Distance

3.3 Experiment Steps

This part makes a summary of the steps in the experiment process for both classification and regression prediction. As shown in the graph3.22, at the first step, we read the dataset. That is, for the classification on 'project_id', 'Standup_final_2' is imported. For the regression on 'time', the dataset 'all_standups_a'is imported. Then, we form the training dataset, for the classification of the 'project_id', the training dataset is a 5027 ×4 matrix, since there are totally 5027 different rows in the dataset and we make a classification on 'project_id' with the use of four features 'user_id', 'members', 'mentions' and 'message_number'. Concerning the prediction of 'time', the training dataset is a 15005 ×3 matrix, since there are totally 15005 different rows in the dataset 'all_standups_a' and we use the three features 'user_id', 'project_id' and 'interest_id' to predict the output 'time'. Then, we make a data normalization for all the data. At the fourth step, we set some values of the parameters for the different algorithms and append the output into the training dataset matrix. Subsequently, we shuffle the data to make the training process random. After this, we make an allocation of the training dataset and the test dataset, in this project, since the data is not very large and to make sure the experiment result has a higher accuracy, we firstly try to use 90% of the data as the training data and 10% as the test data. Also, we try an allocation of the dataset with 80% of the data as the training data and 20% as the test data to make a com-

parison. Then, we running the algorithms and make a evaluation of the results lastly.

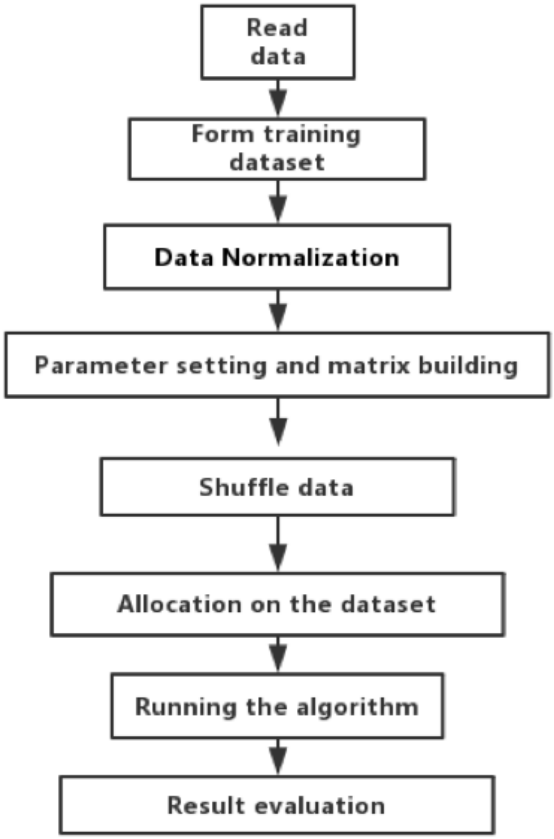


Figure 3.22: Experiment Process

Chapter 4

Presentation and Evaluation of Results

This chapter makes a summary of the experiment results with an evaluation. The results show that, for the classification on 'project_id', the multiple neural network algorithm can get a best result with label powerset method when the number of nodes per layer is 25 and the percentage of test dataset is 20%. For the prediction on 'time', the random forest regression is the best when the percentage of test dataset is 20%.

4.1 Classification on Project_id

For the result of classification on 'project_id', we use the Classifier Chain and the Label Powerset methods which are mentioned in literature review to simplify the multi-label classification problem. We measure the accuracy in terms of the percentage. Firstly, consider the Gaussian Naive Bayesian algorithm, we conduct the experiment for 10 times and take an average of the accuracy results with multi-label classification methods of classifier chain and label powerset respectively . As shown in the table 4.1, when the percentage of test dataset is 10%, the average accuracy when using classifier chain is only 4.42% which is significantly lower than that when using label powerset (13.38%). From table 4.2, when the percentage of test dataset is 20%, the average accuracy when using classifier chain is 5.57%

which is also significantly lower than that when using label powerset (14.22%).

Experiment number	Classifier Chain	Label Powerset
1	0.0319	0.1335
2	0.0545	0.1354
3	0.0498	0.1348
4	0.0389	0.1326
5	0.0425	0.1299
6	0.0356	0.1335
7	0.0455	0.1425
8	0.0366	0.1385
9	0.0543	0.1342
10	0.0526	0.1233
Average	0.0442	0.1338

Table 4.1: Gaussian Naive Bayesian Results with 10% Test Data

Experiment number	Classifier Chain	Label Powerset
1	0.0457	0.1293
2	0.0497	0.1383
3	0.0507	0.1353
4	0.0477	0.1343
5	0.0636	0.1661
6	0.0547	0.1542
7	0.0557	0.1422
8	0.0517	0.1273
9	0.0517	0.1283
10	0.0398	0.1422
Average	0.0511	0.1398

Table 4.2: Gaussian Naive Bayesian Results with 20% Test Data

For the multiple neural network algorithm, under the condition that the num-

ber of hidden layers (L) is set at 2, a learning rate (R) is set at 0.1 and the iteration is set to be 1000, we change the number of nodes per layer (N) from 5 to 50 to see which value of nodes per layer get the most accurate result. When the percentage of test dataset is 10%, the result is shown in table 4.3. From the table, we can see that classifier chain method gets the highest accuracy (27.69%) with N=35. For label powerset method, the highest accuracy (29.08%) is obtained when N is 15 or 50. When we present these results in figure 4.1, it is obvious that Label Powerset method has a higher accuracy in general. Therefore, when the percentage of test dataset is 10%, the multiple neural network algorithm can get a best result with label powerset method when the number of nodes per layer is 15 or 50.

number of nodes per layer (N)	Classifier Chain	Label Powerset
5	0.2092	0.1972
10	0.2390	0.2331
15	0.2470	0.2908
20	0.2410	0.2789
25	0.2311	0.2849
30	0.2649	0.2789
35	0.2769	0.2709
40	0.2530	0.2769
45	0.2231	0.2490
50	0.2696	0.2908

Table 4.3: Multiple Neural Network Results with 10% Test Data

When the percentage of test dataset is 20%, the result is shown in table 4.4. From the table, we can see that a classifier chain method gets the highest accuracy (27.76%) when N=25 or N=50. For label powerset method, the highest accuracy (31.04%) is obtained when N is 25. When we present these results in figure 4.2, it is obvious that Label Powerset method has a higher accuracy in general. Therefore, when the percentage of test dataset is 20%, the multiple neural network algorithm

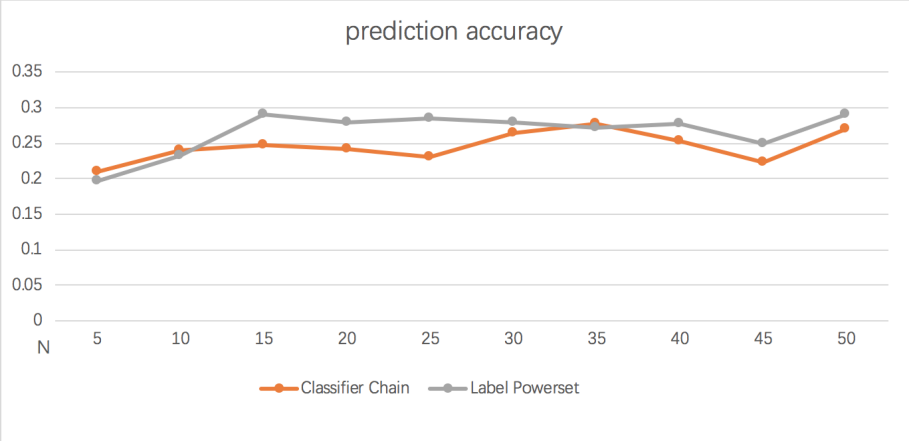


Figure 4.1: Prediction Accuracy

number of nodes per layer (N)	Classifier Chain	Label Powerset
5	0.1462	0.2388
10	0.2557	0.2577
15	0.2427	0.2457
20	0.2467	0.2606
25	0.2776	0.3104
30	0.2636	0.2905
35	0.2766	0.2855
40	0.2507	0.2805
45	0.2686	0.2935
50	0.2776	0.2776

Table 4.4: Multiple Neural Network Results with 20% Test Data

can get a best result with label powerset method when the number of nodes per layer is 25.

Overall, concerning the Gaussian Naive Bayesian algorithm, by using a label powerset method, it has a highest accuracy (14.22%) when the percentage of test dataset is 20%. For the multiple neural network algorithm, by using label powerset method, it has a highest prediction accuracy 31.04% when the percentage of test dataset is 20% with the number of nodes per layer at 25. Therefore, we can

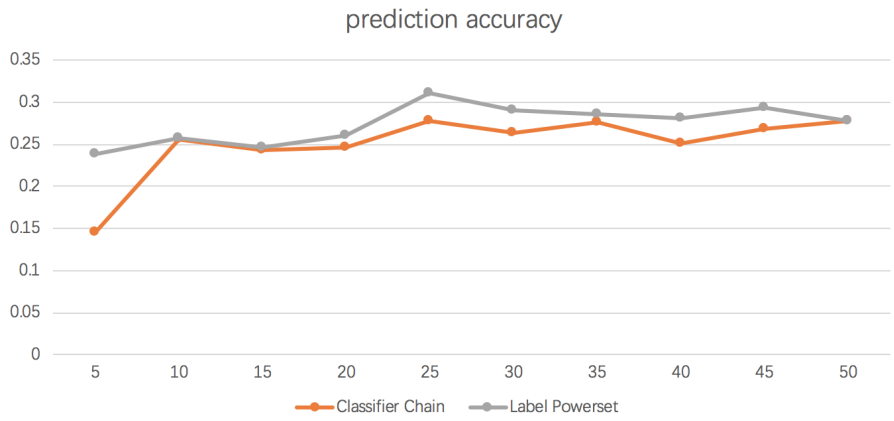


Figure 4.2: Prediction Accuracy

conclude that the multiple neural network algorithm is better than the Gaussian Naive Bayesian algorithm concerning the classification on 'project_id'. However, the accuracy 31.04% is significantly low in general for the classification result. The reason for this can explained by the datasets. The information from all the six datasets 'all_standups', 'Slack_Public' 'Slack_Private', 'User_mapping'. 'Interest_group_info' and 'Channel_info' is not very clearly presented. Also, after the data processing, there are only 5027 rows of data information available, which is not large enough for classification.

4.2 Prediction for Time

For the prediction on 'time', firstly we look at the result of the predicted value on 'time' which has a unit of minutes. We use an example result to explain. We try to predict the value of 'time' when the 'user_id' is set to be 6 with 'project_id' to be 120 and 'interest_id' to be 37. Firstly, considering the circumstance when the percentage of test dataset is 10%, by changing the parameter which is the number of power in polynomial regression algorithm from 1 to 10, we get the results for three models 'polynomial regeression', 'random forest regression' and 'SVM regression' as shown in the table 4.5. We present these results in the figure 4.3 , from the dataset 'all_standups_a', the original value of 'time' is 120 minutes when the 'user_id' 'project_id' and 'interest_id' are 6, 120 and 37 respectively. We want

to select the model which can get the predicted value that has the smallest error from the original value of '120'. From the figure 4.4, we can see that in general, the orange line which represents the polynomial regression has a large distance with the horizontal line of 120. The yellow line is most fitted with the horizontal line of 120. That means SVM regression has the best fitting with the data. More specifically, when number of power is 6. The error of the SVM regression model is only 0.023 (120.023-120). Therefore, SVM regression is the best one among these three models for the predicted value.

number of power in polynomial regression algorithm	Polynomial regression	Random forest regression	SVM regression
1	187.414	113.168	116.652
2	146.205	104.930	119.830
3	153.231	113.697	116.151
4	171.328	114.248	119.374
5	132.802	110.246	119.535
6	142.025	117.257	120.023
7	138.527	110.914	119.821
8	144.389	109.422	120.754
9	115.826	111.540	115.990
10	139.673	116.780	116.540

Table 4.5: Polynomial Regression Results with 10% Test Data

Then, concerning the error of prediction, we use the mean squared error (MSE) to evaluate. For the case when the percentage of test dataset is 10%. The table 4.7 below shows the result. We also presents these results in figure 4.5a , it is obvious that the gray line is significantly lower than the orange and yellow line. That is, in general, the MSE of random forest regression is the smallest. Also, when the number of power is 2, the MSE of random forest regression is optimal.

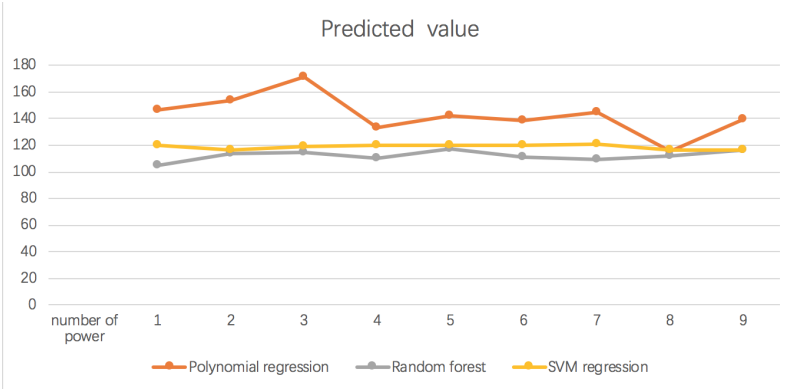


Figure 4.3: Prediction Value with 10% Test Data

number of power in polynomial regression algorithm	Polynomial regression	Random forest regression	SVM regression
1	185.838	98.490	109.774
2	149.779	118.754	119.106
3	150.955	107.022	114.130
4	170.162	108.317	119.627
5	126.991	111.988	121.289
6	152.989	111.988	121.289
7	141.393	122.469	121.504
8	129.221	110.686	121.452
9	118.192	114.024	122.513
10	153.316	115.510	122.251

Table 4.6: Polynomial Regression Results with 10% Test Data

Similarly, when the percentage of test dataset is 20%, from the table 4.8 and figure 4.5b , we can also see that the MSE of random forest regression is significantly lower than SVM and polynomial regression algorithm. Besides, the overall MSE is significantly lower than that in the case of 10% test dataset.

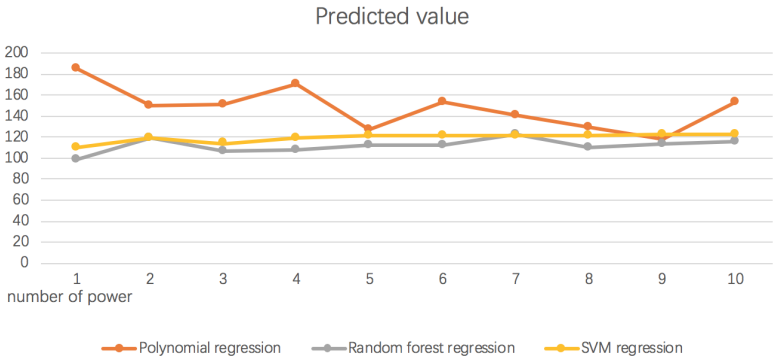


Figure 4.4: Prediction Value with 20% Test Data

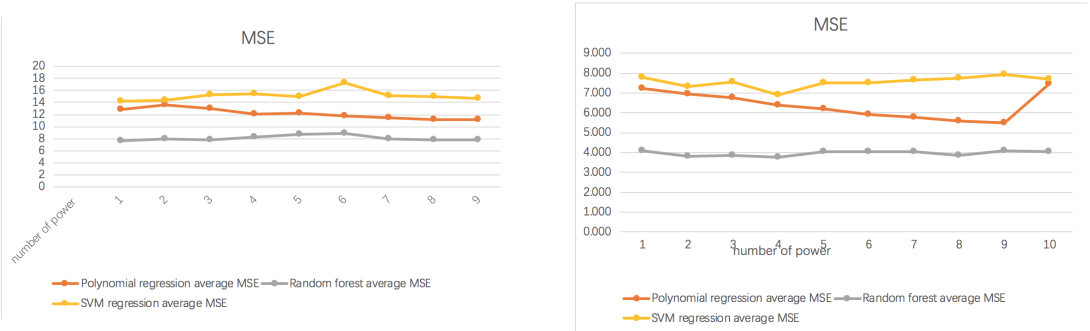
number of power in polynomial regression algorithm	Polynomial regression average MSE	Random forest average MSE	SVM regression average MSE
1	13.424	7.981	14.589
2	12.918	7.609	14.296
3	13.607	8.000	14.445
4	13.015	7.853	15.344
5	12.039	8.199	15.481
6	12.250	8.772	14.925
7	11.750	8.928	17.254
8	11.540	7.898	15.088
9	11.212	7.789	14.975
10	11.231	7.775	14.655

Table 4.7: MSE Evaluation

Combining the results of predicted value and MSE evaluation together, we can conclude that the random forest regression is the best to predict 'time' among these three algorithms because it has a significant lower MSE compared with other two algorithms. In other words, it has a lowest prediction error. Besides, it is also robust for the predicted value prediction although it is not such fitted to the data as SVM regression.

number of power in polynomial regression algorithm	Polynomial regression average MSE	Random forest average MSE	SVM regression average MSE
1	7.214	4.077	7.770
2	6.953	3.808	7.327
3	6.754	3.855	7.552
4	6.373	3.758	6.921
5	6.180	4.037	7.505
6	5.929	4.037	7.505
7	5.762	4.058	7.653
8	5.579	3.873	7.739
9	5.520	4.109	7.942
10	7.456	4.037	7.698

Table 4.8: MSE Evaluation



(a) MSE Value with 10% Test Data

(b) MSE Value with 20% Test Data

Figure 4.5: MSE Value with Different Percent of Test Data

Chapter 5

Conclusion and Further Work

In conclusion, this project aims at solving the problem of 'stand_up' in Satalia by making a prediction on the 'stand_up' data. In order to achieve this objective, multiple neural network and Gaussian Naive Bayesian model are used to make a classification on the output 'project_id with 'user_id', 'members', 'mentions' and 'message_number' as inputs. Besides, we make a prediction on the 'time' by using the employee's 'user_id','project_id' and 'interest_id' with polynomial regression, random forest regression and Support vector machine (SVM) regression as the algorithms. The empirical results show that in general, for the classification on 'project_id', the multiple neural network algorithm can get a best result with label powerset method. For the prediction on 'time', the random forest regression is the best. However, the result from the classification is not accurate enough, which can be explained by the problem of the datasets. In practice, the result from the classification on 'project_id' is not robust enough to be used in Satalia since its low accuracy while the prediction result on 'time' can be utilized for Satalia since it has a relatively high accuracy. There are some further works can be done to improve the result of this research. Firstly, the dataset should be complete enough and the content of the dataset should be clear enough for prediction. Besides, the dataset should be larger, then we can obtain a result with a higher accuracy. Last but not least, we should also try to use some other algorithms to obtain the results and then make a comparison with the results obtained from the algorithms used in this project.

Chapter 6

Learning Points

Embarking on this project has enabled acquisition of a much better understanding on how to conduct a research project and thinking of feasible solutions to solve problems in the field of computer science and technology. By taking advice from the project supervisor and reading many related papers, most risks involved with carrying out a project that involves lacking of hardware for training and new problem proposed for strengthen this project were eliminated.

In this project, I learned several courses, Machine Learning taught by Andrew Ng [51], Neural networks for machine learning [28] taught by Geoffrey Hinton on the website of Coursera and Cs231n: Convolutional neural networks for visual recognition [35] on Youtube. Through these courses, I learned a lot about the basic knowledge of machine learning such as polynomial regression, neural networks, Random Forest Regression, SVM(Supported Vector Machine) and Gaussian Naive Bayesian, and also different kinds of neural network like CNN, RNN and their structure in details.

Chapter 7

Professional Issues

According to the code of practice and conduct issued by the British Computer Society, my project was finished punctually and I can confirm the purpose that, the whole system I designed is to benefit society. During the process of this project I always following the “Key IT practices”, for example:

1. Offering complete information on the project’s milestones and outcomes;

2. Regularly doing revision for avoiding risks and reporting any key issues that influence the design and implementation of project;

3. Finding out relative projects to study better researching methods and trying to eliminate issues they have encountered before;

4. Timely and frankly summarizing problems encountered, and knowledge learned.

As for the code of wrote, public’s safety, wellness, and environment are all considered. As stated in the code of conduct, I shall: 1. Report all the possible risks and results;

2. Regard the legal rights of third parties, project supervisor, and professional peers;

3. Carry out activities in a professional manner without distinctions;

4. Preserve secret information and not expose it for personal benefit unless the requirement or permission from the involved authority or in a court of law.

Bibliography

[1] Ajith Abraham. “Artificial neural networks”. In: *handbook of measuring system design* (2005).

[2] Ajith Abraham. “Artificial neural networks”. In: *handbook of measuring system design* (2005).

[3] Noura AlDarmaki et al. “Prediction of the closing price in the Dubai financial market: A data mining approach”. In: *Big Data and Smart City (ICBDSC), 2016 3rd MEC International Conference on*. IEEE. 2016, pp. 1–7.

[4] Leemon Baird. “Residual algorithms: Reinforcement learning with function approximation”. In: *Machine Learning Proceedings 1995*. Elsevier, 1995, pp. 30–37.

[5] Benjamin M Bolstad et al. “A comparison of normalization methods for high density oligonucleotide array data based on variance and bias”. In: *Bioinformatics* 19.2 (2003), pp. 185–193.

[6] Nataliya Boyko, Tetyana Sviridova, and Nataliya Shakhovska. “Use of machine learning in the forecast of clinical consequences of cancer diseases”. In: *2018 7th Mediterranean Conference on Embedded Computing (MECO)*. IEEE. 2018.

[7] F Jay Breidt and Jean D Opsomer. “Local polynomial regression estimators in survey sampling”. In: *Annals of Statistics* (2000), pp. 1026–1053.

[8] Rodney A Brooks. “Intelligence without representation”. In: *Artificial intelligence* 47.1-3 (1991), pp. 139–159.

[9] Rich Caruana and Alexandru Niculescu-Mizil. “An empirical comparison of supervised learning algorithms”. In: *Proceedings of the 23rd international conference on Machine learning*. ACM. 2006, pp. 161–168.

[10] Rich Caruana and Alexandru Niculescu-Mizil. “An empirical comparison of supervised learning algorithms”. In: *Proceedings of the 23rd international conference on Machine learning*. ACM. 2006, pp. 161–168.

[11] Jeffrey Cleveland et al. “Scalable machine learning framework for behavior-based access control”. In: *Resilient Control Systems (ISRCS), 2013 6th International Symposium on*. IEEE. 2013, pp. 181–185.

[12] Christian W Dawson and Robert Wilby. “An artificial neural network approach to rainfall-runoff modelling”. In: *Hydrological Sciences Journal* 43.1 (1998), pp. 47–66.

[13] Krzysztof Dembszynski et al. “On label dependence in multilabel classification”. In: *LastCFP: ICML Workshop on Learning from Multi-label data*. Ghent University, KERMIT, Department of Applied Mathematics, Biometrics and Process Control. 2010.

[14] Thomas G Dietterich. “Approximate statistical tests for comparing supervised classification learning algorithms”. In: *Neural computation* 10.7 (1998), pp. 1895–1923.

[15] Ramón Dıaz-Uriarte and Sara Alvarez De Andres. “Gene selection and classification of microarray data using random forest”. In: *BMC bioinformatics* 7.1 (2006), p. 3.

[16] Pedro Domingos and Michael Pazzani. “On the optimality of the simple Bayesian classifier under zero-one loss”. In: *Machine learning* 29.2-3 (1997), pp. 103–130.

[17] Harris Drucker et al. “Support vector regression machines”. In: *Advances in neural information processing systems*. 1997, pp. 155–161.

[18] Morris L Eaton. *The Gauss-Markov theorem in multivariate analysis*. Tech. rep. University of Minnesota, 1983.

[19] Jeffrey R Edwards and Mark E Parry. “On the use of polynomial regression equations as an alternative to difference scores in organizational research”. In: *Academy of Management Journal* 36.6 (1993), pp. 1577–1613.

[20] Kunihiko Fukushima and Sei Miyake. “Neocognitron: A self-organizing neural network model for a mechanism of visual pattern recognition”. In: *Competition and cooperation in neural nets*. Springer, 1982, pp. 267–285.

[21] Paul Geladi and Bruce R Kowalski. “Partial least-squares regression: a tutorial”. In: *Analytica chimica acta* 185 (1986), pp. 1–17.

[22] David E Goldberg and John H Holland. “Genetic algorithms and machine learning”. In: *Machine learning* 3.2 (1988), pp. 95–99.

[23] Ulrike Grömping. “Variable importance assessment in regression: linear regression versus random forest”. In: *The American Statistician* 63.4 (2009), pp. 308–319.

[24] Steve R Gunn et al. “Support vector machines for classification and regression”. In: *ISIS technical report* 14.1 (1998), pp. 5–16.

[25] Lars Kai Hansen and Peter Salamon. “Neural network ensembles”. In: *IEEE transactions on pattern analysis and machine intelligence* 12.10 (1990), pp. 993–1001.

[26] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. “Unsupervised learning”. In: *The elements of statistical learning*. Springer, 2009, pp. 485–585.

[27] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. “Unsupervised learning”. In: *The elements of statistical learning*. Springer, 2009, pp. 485–585.

[28] Geoffrey Hinton, Nitsh Srivastava, and Kevin Swersky. “Neural networks for machine learning”. In: *Coursera, video lectures* 264 (2012).

[29] Tin Kam Ho. “Random decision forests”. In: *Document analysis and recognition, 1995., proceedings of the third international conference on*. Vol. 1. IEEE. 1995, pp. 278–282.

[30] Thomas Hofmann. “Unsupervised learning by probabilistic latent semantic analysis”. In: *Machine learning* 42.1-2 (2001), pp. 177–196.

[31] George H John and Pat Langley. “Estimating continuous distributions in Bayesian classifiers”. In: *Proceedings of the Eleventh conference on Uncertainty in artificial intelligence*. Morgan Kaufmann Publishers Inc. 1995, pp. 338–345.

[32] George H John and Pat Langley. “Estimating continuous distributions in Bayesian classifiers”. In: *Proceedings of the Eleventh conference on Uncertainty in artificial intelligence*. Morgan Kaufmann Publishers Inc. 1995, pp. 338–345.

[33] George H John and Pat Langley. “Estimating continuous distributions in Bayesian classifiers”. In: *Proceedings of the Eleventh conference on Uncertainty in artificial intelligence*. Morgan Kaufmann Publishers Inc. 1995, pp. 338–345.

[34] Michael I Jordan. “Supervised learning and systems with excess degrees of freedom”. In: (1988).

[35] Andrej Karpathy. “Cs231n: Convolutional neural networks for visual recognition”. In: *Online Course* (2016).

[36] Ron Kohavi. “Scaling up the accuracy of Naive-Bayes classifiers: a decision-tree hybrid.” In: *KDD*. Vol. 96. Citeseer. 1996, pp. 202–207.

[37] Miroslav Kubat, Robert C Holte, and Stan Matwin. “Machine learning for the detection of oil spills in satellite radar images”. In: *Machine learning* 30.2-3 (1998), pp. 195–215.

[38] Quoc V Le. “Building high-level features using large scale unsupervised learning”. In: *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*. IEEE. 2013, pp. 8595–8598.

[39] Erich L Lehmann and George Casella. *Theory of point estimation*. Springer Science & Business Media, 2006.

[40] Andy Liaw, Matthew Wiener, et al. “Classification and regression by randomForest”. In: *R news* 2.3 (2002), pp. 18–22.

[41] Claudia Lindner et al. “Fully automatic segmentation of the proximal femur using random forest regression voting”. In: *IEEE transactions on medical imaging* 32.8 (2013), pp. 1462–1472.

[42] Claudia Lindner et al. “Robust and accurate shape model matching using random forest regression-voting”. In: *IEEE transactions on pattern analysis and machine intelligence* 37.9 (2015), pp. 1862–1874.

[43] Ian B MacNeill et al. “Properties of sequences of partial sums of polynomial regression residuals with applications to tests for change of regression at unknown times”. In: *The Annals of Statistics* 6.2 (1978), pp. 422–433.

[44] Andrew McCallum, Kamal Nigam, et al. “A comparison of event models for naive bayes text classification”. In: *AAAI-98 workshop on learning for text categorization*. Vol. 752. 1. Citeseer. 1998, pp. 41–48.

[45] Vangelis Metsis, Ion Androutsopoulos, and Georgios Paliouras. “Spam filtering with naive bayes-which naive bayes?” In: *CEAS*. Vol. 17. Mountain View, CA. 2006, pp. 28–69.

[46] D Meyer. *Support vector machines. R News, 1 (3): 23–26*. 2001.

[47] Tom M Mitchell et al. *Machine learning. WCB*. 1997.

[48] Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. *Foundations of machine learning*. MIT press, 2012.

[49] M Narasimha Murty and V Susheela Devi. *Pattern recognition: An algorithmic approach*. Springer Science & Business Media, 2011.

[50] Enric Musoll, Tomás Lang, and Jordi Cortadella. “Working-zone encoding for reducing the energy in microprocessor address buses”. In: *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* 6.4 (1998), pp. 568–572.

[51] Andrew Ng. *Machine Learning. Coursera*. 2016.

[52] Andrew Y Ng and Michael I Jordan. "On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes". In: *Advances in neural information processing systems*. 2002, pp. 841–848.

[53] Martijn van Otterlo and Marco Wiering. "Reinforcement learning and markov decision processes". In: *Reinforcement Learning*. Springer, 2012, pp. 3–42.

[54] S Pratiher, M Mukherjee, and N Haque. "A Multifractal Detrended Fluctuation Analysis-Based Framework for Fault Diagnosis in Autonomous Microgrids". In: *Advances in Communication, Devices and Networking*. Springer, 2018, pp. 199–207.

[55] Jesse Read et al. "Classifier chains for multi-label classification". In: *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer. 2009, pp. 254–269.

[56] Jesse Read et al. "Classifier chains for multi-label classification". In: *Machine learning* 85.3 (2011), p. 333.

[57] Martin Riedmiller. "Advanced supervised learning in multi-layer perceptrons-from backpropagation to adaptive learning algorithms". In: *Computer standards and interfaces* 16.3 (1994), pp. 265–278.

[58] Jürgen Schmidhuber. "Deep learning in neural networks: An overview". In: *Neural networks* 61 (2015), pp. 85–117.

[59] Burkhardt Seifert and Theo Gasser. "Local polynomial smoothing". In: *Encyclopedia of statistical sciences* 7 (2004).

[60] Ingo Steinwart and Andreas Christmann. *Support vector machines*. Springer Science & Business Media, 2008.

[61] Carolin Strobl et al. "Bias in random forest variable importance measures: Illustrations, sources and a solution". In: *BMC bioinformatics* 8.1 (2007), p. 25.

[62] Richard S Sutton and Andrew G Barto. *Introduction to reinforcement learning*. Vol. 135. MIT press Cambridge, 1998.

[63] Richard S Sutton, Andrew G Barto, et al. *Reinforcement learning: An introduction*. MIT press, 1998.

[64] Johan AK Suykens and Joos Vandewalle. "Least squares support vector machine classifiers". In: *Neural processing letters* 9.3 (1999), pp. 293–300.

[65] Vladimir Svetnik et al. "Random forest: a classification and regression tool for compound classification and QSAR modeling". In: *Journal of chemical information and computer sciences* 43.6 (2003), pp. 1947–1958.

[66] Henri Theil. "A rank-invariant method of linear and polynomial regression analysis". In: *Henri Theil's contributions to economics and econometrics*. Springer, 1992, pp. 345–381.

[67] Theodore B Trafalis and Huseyin Ince. "Support vector machine for regression and applications to financial forecasting". In: *Neural Networks, 2000. IJCNN 2000, Proceedings of the IEEE-INNS-ENNS International Joint Conference on*. Vol. 6. IEEE. 2000, pp. 348–353.

[68] Grigorios Tsoumakas and Ioannis Katakis. "Multi-label classification: An overview". In: *International Journal of Data Warehousing and Mining (IJDWM)* 3.3 (2007), pp. 1–13.

[69] Grigorios Tsoumakas and Ioannis Vlahavas. "Random k-labelsets: An ensemble method for multilabel classification". In: *European conference on machine learning*. Springer. 2007, pp. 406–417.

[70] Sun-Chong Wang. "Artificial neural network". In: *Interdisciplinary computing in java programming*. Springer, 2003, pp. 81–100.

[71] Ronald J Williams. "Simple statistical gradient-following algorithms for connectionist reinforcement learning". In: *Machine learning* 8.3-4 (1992), pp. 229–256.

[72] B Yegnanarayana. *Artificial neural networks*. PHI Learning Pvt. Ltd., 2009.

[73] B Yegnanarayana. *Artificial neural networks*. PHI Learning Pvt. Ltd., 2009.

[74] Andreas Zell. *Simulation neuronaler netze*. Vol. 1. Addison-Wesley Bonn, 1994.

[75] Andreas Zell. *Simulation neuronaler netze*. Vol. 1. Addison-Wesley Bonn, 1994.

[76] Harry Zhang. “The optimality of naive Bayes”. In: *AA* 1.2 (2004), p. 3.

Appendix A

Codes Implemented for This Project

A.1 Code for Reading Data and Data-Preprocessing

```
1
2 # coding: utf-8
3
4 # In[26]:
5
6 import csv
7 import tqdm
8
9
10 # In[27]:
11
12 def id_mapping():
13     filename = 'data/metadata/user_mapping.csv'
14     with open(filename) as f:
15         reader = csv.DictReader(f)
16         dic = dict()
17         for row in reader:
18             user_id = row['standup_id']
19             user_name = row['slack_id']
20             dic[user_name] = user_id
21     return dic
22
```

```
23
24 # In[28]:
25
26 def channel_mapping():
27     filename = 'data/metadata/channel_info.csv'
28     with open(filename) as f:
29         reader = csv.DictReader(f)
30         dic = dict()
31         for row in reader:
32             channel_id = row['channel_id']
33             project_id = row['project_id']
34             if project_id != '':
35                 dic[channel_id] = project_id
36     return dic
37
38
39 # In[29]:
40
41 def read_data(filename):
42     id_list = []
43     with open(filename, encoding="ISO-8859-1") as f:
44         reader = csv.DictReader(f)
45         for row in reader:
46             user_id = row['user_id']
47             id_list.append(user_id)
48     return id_list
49
50
51 # In[1]:
52
53 def write_id_data(dic, id_list, filename):
54     #missing_key = set()
55     with open(filename, 'w') as f:
56         writer = csv.writer(f)
57         for id_name in tqdm.tqdm(id_list):
58             if(id_name in dic):
59                 writer.writerow([dic[id_name]])
```

```
60              else:
61                  writer.writerow(['XXXXX'])
62                  #missing_key.add(id_name)
63      #return missing_key
64
65
66  # In[ ]:
67
68  def write_member_data(dic, id_list, filename):
69      with open(filename, 'w') as f:
70          writer = csv,writer(f)
71          for id_name in dqdm.tqdm(id_list):
72
73
74  # In[31]:
75
76  def write_missing_key(missing_key, filename):
77      with open(filename, 'w') as f:
78          writer = csv.writer(f)
79          for key in tqdm.tqdm(missing_key):
80              writer.writerow([key])
81
82
83  # In[3]:
84
85  def main(args):
86      dic_id = id_mapping()
87      dic_channel = channel_mapping()
88
89
90      #----------------------------------------Private Slack
        ---------------------------------
91      if args == 'Private Slack' :
92          filename_private_read = 'data/Slack_Private_19062018.csv'
93          filename_private_write = 'processed_data_private.csv'
94          #filename_private_missing_key = 'missing_key_private.csv'
95
```

```
96          id_list_private = read_data(filename_private_read)
97          write_id_data(dic_id, id_list_private, filename_private_write)
98          #write_missing_key(missing_key_private,
    filename_private_missing_key)
99
100
101
102
103     #--------------------------------------Public Slack
    ------------------------------
104     if args == 'Public Slcak' :
105         filename_public_read = 'data/Slack_Public_19062018.csv'
106         filename_public_write = './data/processed_data_public.csv'
107         #filename_public_missing_key = 'missing_key_public.csv'
108         dic = id_mapping()
109         id_list_public = read_data(filename_public_read)
110         write_id_data(dic, id_list_public, filename_public_write)
111         #write_missing_key(missing_key_public,
    filename_public_missing_key)
112
113
114
115
116     #--------------------------------------Channel Setting
    ------------------------------
117     if args == 'Channel Setting':
118         filename_channel_read = 'data/Slack_Public_19062018.csv'
119         filename_channel_write = 'data/precessed_channel_data.csv'
120
121
122
123 # In[33]:
124
125 if __name__ == "__main__":
126     args = 'Private Slack'
127     main(args)
128
```

```
129
130 # In[ ]:
```

A.2 Time Prediction Code

```
1
2 # coding: utf-8
3
4 # In[15]:
5
6 import numpy as np
7 import csv
8 import matplotlib.pyplot as plt
9 import tqdm
10 import math
11 from sklearn.ensemble import RandomForestRegressor
12 from sklearn.svm import SVR
13 import matplotlib.pyplot as plt
14 from mpl_toolkits.mplot3d import Axes3D
15 from sklearn.preprocessing import StandardScaler
16 from sklearn.preprocessing import PolynomialFeatures
17 from sklearn.metrics import mean_squared_error
18
19
20 # In[16]:
21
22 poly = PolynomialFeatures(2)
23 scaler = StandardScaler()
24
25
26 # In[9]:
27
28 def form_data():
29     filename = 'data/all_standups_jun17_jun18.csv'
30     user_id = np.zeros((15005, 1))
31     interest_id = np.zeros((15005, 1))
32     project_id = np.zeros((15005, 1))
```

```
33      time = np.zeros((15005, 1))
34      index = 0;
35      with open(filename) as f:
36          reader = csv.DictReader(f)
37          for row in reader:
38              user_id[index, :] = row['user_id']
39              interest_id[index, :] = row['interest_id']
40              project_id[index, :] = row['project_id']
41              time[index, :] = row['time']
42              index += 1
43
44      n = time.shape[0]
45      data = np.zeros((n, 3))
46      data[:, 0] = user_id[:,0]
47      data[:, 1] = interest_id[:,0]
48      data[:, 2] = project_id[:,0]
49
50      data = poly_features(data)
51      data = normalization(data)
52      print(data.shape)
53      data = np.append(data, time, axis=1)
54      np.random.shuffle(data)
55      print(data.shape)
56      return data
57
58
59  # In[10]:
60
61  def normalization(data):
62      scaler.fit(data)
63      trans_data = scaler.transform(data)
64      return trans_data
65
66
67  # In[11]:
68
69  def poly_features(data):
```

```
70      dataTransform = poly.fit_transform(data)
71      return dataTransform
72
73
74  # In[ ]:
75
76  def linear_regression_model(train, test):
77      model = LinearRegression(normalize = True)
78      label_length = train.shape[1] - 1
79      x = train[:, 0:label_length]
80      y = train[:, label_length]
81      model.fit(x, y)
82      #for i in range(test.shape[0]):
83          #print(test[i, label_length].reshape(-1, 1) - model.predict(
        test[i, 0:label_length]).reshape(-1, 1))
84      error = np.sum(mean_squared_error(test[:, label_length], model.
        predict(test[:, 0:label_length]))) / test.shape[0]
85      return error, model
86
87
88  # In[34]:
89
90  def radom_forest_regressor(train, test):
91      model = RandomForestRegressor()
92      label_length = train.shape[1] - 1
93      x = train[:, 0:label_length]
94      y = train[:, label_length]
95      model.fit(x, y)
96      #for i in range(test.shape[0]):
97          #print(test[i, label_length].reshape(-1, 1) - model.predict(
        test[i, 0:label_length]).reshape(-1, 1))
98      error = np.sum(mean_squared_error(test[:, label_length], model.
        predict(test[:, 0:label_length]))) / test.shape[0]
99      return error, model
100
101
102 # In[35]:
```

```
103
104 def SVM_regressor(train, test):
105     model = SVR()
106     label_length = train.shape[1] - 1
107     x = train[:, 0:label_length]
108     y = train[:, label_length]
109     model.fit(x, y)
110     #for i in range(test.shape[0]):
111         #print(test[i, label_length].reshape(-1, 1) - model.predict(
    test[i, 0:label_length]).reshape(-1, 1))
112     error = np.sum(mean_squared_error(test[:, label_length], model.
    predict(test[:, 0:label_length]))) / test.shape[0]
113     return error, model
114
115
116 # In[36]:
117
118 def form_test_train(data, n):
119     test_size = math.floor(n * 0.1)
120     test = np.zeros((test_size, data.shape[1]))
121     train = np.zeros((n - test_size, data.shape[1]))
122     test = data[0:test_size, :]
123     train = data[test_size:, :]
124     return test, train
125
126
127 # In[37]:
128
129 def data_pre_process():
130     data = form_data()
131     test, train = form_test_train(data, data.shape[0])
132     return train, test
133
134
135 # In[44]:
136
137 def single_test(data, model):
```

```
    print('test normalized data: ' ,data.shape)
    return model.predict(data)


# In[47]:

def main():
    train, test = data_pre_process()

    Poly_error, polynomial_model = radom_forest_regressor(train, test)
    print('Random Forest Average Error: ', Foreast_error)
    Foreast_error, random_forest_model = radom_forest_regressor(train,
    test)
    print('Random Forest Average Error: ', Foreast_error)
    SVR_errors, SVR_model = SVM_regressor(train, test)
    print('SVM Regression Average Error: ', SVR_errors)

    x = poly.transform(np.array([3, 29, 2]).reshape(1, -1).astype(float
    ))
    x = scaler.transform(x)
    #print(x)
    #print(test[0,:])
    print(single_test(x, random_forest_model))


# In[48]:

if __name__ == "__main__":
    main()


# In[ ]:
```

A.3 Project ID Prediction Code

```

# coding: utf-8
```

```
3
4 # In[60]:
5
6 import csv
7 import numpy as np
8 import math
9 from skmultilearn.problem_transform import BinaryRelevance
10 from sklearn.naive_bayes import GaussianNB
11 from sklearn.metrics import accuracy_score
12 from skmultilearn.problem_transform import ClassifierChain
13 from skmultilearn.problem_transform import LabelPowerset
14 from skmultilearn.adapt import MLkNN
15 from sklearn.neural_network import MLPClassifier
16 from sklearn.ensemble import RandomForestClassifier
17 from sklearn.svm import LinearSVC
18
19
20 # In[2]:
21
22 def get_dict(filename, key, value):
23     with open(filename) as f:
24         reader = csv.DictReader(f)
25
26         #create map
27         dict_ = dict()
28
29         for row in reader:
30             key_val = row[key]
31             val_val = row[value]
32             dict_[int(key_val)] = int(val_val)
33
34         return dict_
35
36
37 # In[3]:
38
39 project_map = get_dict('data/encoding/project_id_encoding.csv', '
```

```
    project_id', 'mapping_number')
user_map = get_dict('data/encoding/user_id_encoding.csv', 'user_id', '
    number_map')
print(len(user_map))
print(len(project_map))


# In[4]:

def read_data(filename):
    with open(filename) as f:
        reader = csv.DictReader(f)

        #create required collections
        project_id = list()
        user_id = list()
        members = list()
        mentions = list()
        message_number = list()

        for row in reader:
            project_id.append(row['project_id'].rstrip().split(' '))
            user_id.append(row['user_id'].rstrip().split(' '))
            members.append(row['members'].rstrip().split(' '))
            mentions.append(row['mentions'].rstrip().split(' '))
            message_number.append(row['message_number'].rstrip().split(
    ' '))
        if '' in members:
            print('True')
            members.remove('')
        if ' ' in mentions:
            print('True')
            mentions.remove('')
        return project_id, user_id, members, mentions, message_number


# In[5]:
```

```
74
75 def get_data_list(filename):
76     project_id, user_id, members, mentions, message_number = read_data(
   filename)
77     return project_id, user_id, members, mentions, message_number
78
79
80 # In[6]:
81
82 def transform_data(data_list, dict_):
83     one_hot_list = np.zeros((len(data_list), len(dict_)))
84     index = 0
85     for sub_list in data_list:
86         one_hot_encode = np.zeros((len(dict_),))
87         row_number = 1
88         for single_data in sub_list:
89             if single_data == '0':
90                 continue
91             if single_data != '':
92                 one_hot_encode[dict_[int(single_data)] - 1] = 1
93         row_number += 1
94         one_hot_list[index, :] = one_hot_encode
95         index += 1
96     return one_hot_list
97
98
99 # In[7]:
100
101 def count_message(message_list):
102     sum_list = np.array((message_list))
103     index = 0
104     for sub_list in message_list:
105         if '' in sub_list:
106             sub_list.remove('')
107         sub_list = list(map(int, sub_list))
108         sum_list[index] = sum(sub_list)
109         index += 1
```

```
110     return sum_list
111
112
113 # In[8]:
114
115 def get_label(project_id, project_map):
116     return transform_data(project_id, project_map)
117
118
119 # In[9]:
120
121 def get_user_id_data(user_id, user_map):
122     return transform_data(user_id, user_map)
123
124
125 # In[10]:
126
127 def get_member_data(members, user_map):
128     for sub_list in members:
129         if '1' in sub_list:
130             sub_list.remove('1')
131         elif '160' in sub_list:
132             sub_list.remove('160')
133         elif '' in sub_list:
134             sub_list.remove('')
135     members_data = transform_data(members, user_map)
136     return members_data
137
138
139 # In[11]:
140
141 def get_mention_data(mentions, user_map):
142     for sub_list in mentions:
143         if '1' in sub_list:
144             sub_list.remove('1')
145         elif '160' in sub_list:
146             sub_list.remove('160')
```

```
147     mentions_data = transform_data(mentions, user_map)
148     return mentions_data
149
150
151 # In[12]:
152
153 def form_data(filename):
154     project_id, user_id, members, mentions, message_number =
    get_data_list(filename)
155     label_data = get_label(project_id, project_map)
156     user_id_data = get_user_id_data(user_id, user_map)
157     member_data = get_member_data(members, user_map)
158     mentions_data = get_mention_data(mentions, user_map)
159     message_number = count_message(message_number)
160     print('Size of user_id: ', user_id_data.shape)
161     print('Size of member_data: ', member_data.shape)
162     print('Size of mentions_data: ', mentions_data.shape)
163     print('Size of message_number: ', message_number.shape)
164     train_data = np.zeros((user_id_data.shape[0], user_id_data.shape[1]
    + member_data.shape[1] + mentions_data.shape[1] +  1))
165     train_data[:, 0:user_id_data.shape[1]] = user_id_data
166     train_data[:, user_id_data.shape[1]:user_id_data.shape[1] +
    member_data.shape[1]] = member_data
167     train_data[:, user_id_data.shape[1] + member_data.shape[1] :
    train_data.shape[1] - 1] = mentions_data
168     train_data[:, train_data.shape[1] - 1] = message_number
169     return train_data, label_data
170
171
172 # In[13]:
173
174 def form_train_test_data(train_data, label_data):
175     data = np.zeros((train_data.shape[0], train_data.shape[1] +
    label_data.shape[1]))
176     data[:, 0:label_data.shape[1]] = label_data
177     data[:,label_data.shape[1]:] = train_data
178     np.random.shuffle(data)
```

```
    n_test = math.floor(0.1 * data.shape[0])
    print('Number of test: ', n_test)
    test = data[0:n_test,:]
    train = data[n_test:,:]
    return train, test


# In[14]:

train_data, label_data = form_data('data/Final_file.csv', )
print(train_data.shape)
print(label_data.shape)
train, test = form_train_test_data(train_data, label_data)


# Binary Relevance, Classifier Chain and Label Powerset are used to transform the multi_label problem to a single label problem.
#
#
# |                          | Binary Relevance | Classifier Chain | Label Powerset |
# |--------------------------|------------------ | ----------------- | ---------------- |
# |  Guassian Naive Bayesian  |  0.0319          |      0.0319      |  0.1335       |
# |  Multiple Neural Network  |  0.253           |       0.255      |  0.1375       |
# |  Random Forest            |   0.263          |      0.247       |  0.269        |
#
#


# In[67]:

Y_train = train[:, 0:label_data.shape[1]]
X_train = train[:,label_data.shape[1]:]

```

```
210 Y_test = test[:, 0:label_data.shape[1]]
211 X_test = test[:,label_data.shape[1]:]
212
213
214 # ### Gaussian Naive Bayesian classification + Binary Relevance
215
216 # In[68]:
217
218 classifier = BinaryRelevance(GaussianNB())
219
220 classifier.fit(X_train, Y_train)
221
222 predictions = classifier.predict(X_test)
223
224 accuracy_score(Y_test, predictions)
225
226
227 # ### Multiple Neural Network + Binary Relevance
228
229 # In[45]:
230
231 mlp = MLPClassifier(solver='lbfgs', activation='relu',alpha=1e-4,
    hidden_layer_sizes=(50,50), random_state=1,max_iter=1000,verbose
    =10,learning_rate_init=.1)
232
233 classifier = BinaryRelevance(mlp)
234
235 classifier.fit(X_train, Y_train)
236
237 predictions = classifier.predict(X_test)
238
239 accuracy_score(Y_test, predictions)
240
241
242 # ### Gaussian Naive Bayesian + Classifier Chain
243
244 # In[26]:
```

```
245
246 classifier = ClassifierChain(GaussianNB())
247
248 classifier.fit(X_train, Y_train)
249
250 predictions = classifier.predict(X_test)
251
252 accuracy_score(Y_test, predictions)
253
254
255 # ### Neural Network + Classifier Chain
256
257 # In[44]:
258
259 mlp = MLPClassifier(solver='lbfgs', activation='relu',alpha=1e-4,
    hidden_layer_sizes=(50,50), random_state=1,max_iter=1000,verbose
    =10,learning_rate_init=.1)
260
261 classifier = ClassifierChain(mlp)
262
263 classifier.fit(X_train, Y_train)
264
265 predictions = classifier.predict(X_test)
266
267 accuracy_score(Y_test, predictions)
268
269
270 # ### Gaussian Naive Bayesian + Label Powerset
271
272 # In[29]:
273
274 classifier = LabelPowerset(GaussianNB())
275
276 classifier.fit(X_train, Y_train)
277
278 predictions = classifier.predict(X_test)
279
```

```
280 accuracy_score(Y_test, predictions)
281
282
283 # ### Neural Network + Label Powerset
284
285 # In[43]:
286
287 mlp = MLPClassifier(solver='lbfgs', activation='relu',alpha=1e-4,
        hidden_layer_sizes=(50,50), random_state=1,max_iter=1000,verbose
        =10,learning_rate_init=.1)
288
289 classifier = LabelPowerset(mlp)
290
291 classifier.fit(X_train, Y_train)
292
293 predictions = classifier.predict(X_test)
294
295 accuracy_score(Y_test, predictions)
```