INDIVIDUAL REPORT

REFLECTIVE LOG

The main objective of this task is to identify a suitable candidate applying for care-related jobs in **HC-One** which is one of the largest care home operators in the UK. The solution is to develop a scoring system that will predict how suitable the candidate is for the given role. This score will help the recruiting team prioritize which candidate gets worked on first. This could save a lot of resources like manpower, money, and time by disregarding some of the intermediate stages of recruiting.

The data given for modelling is taken from websites like indeed, CV-library, etc. Due to the confidential nature of the data we did not use Git version control and special attention is given while processing the data. There were mainly three files, which represent the candidate's application details, educational background, and employment history respectively for the past 6 years. The file which contains the candidate’s application details was considered for modelling since these values are relatively more important and also due to the high quality of this data. The data indicate two types of information, one is internal in nature which means it is specific to a particular job role and the candidate, eg, job title, Advertised salary, etc, and the other one is external in nature which shows the candidate’s catchment and demographic information.

Our team consists of seven members selected randomly for completing this task. On the first day itself, we together figured out the meaning of all the columns and how it will impact our modelling part. Additionally, we formulated a plan for completing this task, such as dividing the work into, sub-task like Exploratory Data Analysis(EDA), preprocessing, scoring method, modelling, visualization, etc. However, even after a brainstorming discussion, the method of creating the scoring system was not found. So, we decided to start our work with EDA hoping that it would give us an idea about the modelling part. So, each person is given some columns to perform EDA. I worked on columns under the categories, *Gender-split in the catchment*, and *relationship status in the catchment*. Then we understood that some of the fields are not necessary as there are many invalid values in certain columns and some columns themselves are irrelevant. This helped us to get an idea of feature selection in advance. The next step was pre-processing of the data which was started by two persons. However, cleaning the salary column was a tricky part so, I volunteered to clean that column alone. The output of the data is fed for feature selection, we used supervised and unsupervised methods as well. To reduce the columns’ count further, I tried Principle Component Analysis (PCA). Although, they were not at all satisfactory. In the drop-in session, we were informed that the length of service can be used as the dependent variable and it can represent the success score. With this in mind, we performed feature selection using the Recursive Feature Extraction(RFE) method and selected 13 variables that could help in predicting the length of service using the decision tree as the base model. Then we decided to model in different approaches. The first one is to consider the service length as the target variable and fit a regression model which could predict the service length of the new candidate based on the candidate data. The value predicted can also represent the success score. We built models like random forest regression, neural networks, and linear regression but the r2 score, as well as the prediction result obtained, were not satisfactory. The next method was to predict the job title using a random forest model. Surprisingly the model yielded good results. Apart from this, we tested the model after splitting the data based on the job title, but the result was unacceptable. Then some statistical models were also implemented, and their performance was low. Finally, we transformed the continuous target variable into a categorical one so that accuracy will increase. As expected, good results were obtained. All these models were done by different members of the team. On the next day, we then discussed selecting the best model out of all the models. On the same day itself, we allocated team members to complete, tasks related to the final deliverables of this module.

My contributions were mainly in the modelling area. Additionally, I worked on data cleaning, and feature extraction using PCA as well. I also completed EDA for selected columns and Feature extraction using RFE as well. In Data cleaning, I found that the primary keys for the main file/data are *Candidate id* and *Vacancy id*. But there are duplicate records based on these columns. That is, for the same primary keys combination the values are different only for the *service lengt*h column and *contracted hours*. So, I discussed this challenge with the team, they suggested that the use of maximum and average aggregate functions will solve this. I used SQL API to complete this task and it worked fine in removing the duplicates. Since we initially considered this problem as unsupervised, we had to select the feature as well, So I studied the techniques like Laplesion Eigenmaps, Autoencoders, Uniform manifold Approximation, Random projection, locally linear embedding and PCA. I implemented PCA having 12 components as it is the best fit for this kind of complex data. However, since the problem become supervised I disregarded it. Initially, we ignored the advertised salary column but later we selected it as it seemed important. So, I cleaned this column. However, cleaning this column was very difficult as there are more than 63 different text formats in this field. So, I developed separate code to extract the salary for each row value for this column. For example, there were salaries given in annually as well as hourly, so I had to deal with these two different formats. Apart from this, there were human errors, as well. So I formulated additional codes to handle these special column values as well. I have made certain assumptions for extracting the salary values, that is If the salary is given as an annual wage then it should be converted to hourly payment by dividing it by 2080 assuming the average working hours in a year is 2080. Then I built two machine-learning models using two separate methods. The first model will predict the Job labels (GradientBoostingClassifier) and the second model (BaggingClassifier) will predict the class /score value which quantifies the suitability of the candidate. Surprisingly for both these models, the test accuracy obtained was more than 90%. Since it seemed less probable, I had to deal with this anomaly. So, I discussed it with the team and the demonstrator, and we concluded that either the models are overfitting, or the models are performing extremely well. Another possibility is that there are similarities in training and test data. Since it is test accuracy, the chances of overfitting are negligible. So I assumed that the quality of the data is poor and took measures to reduce the chances of overfitting as well. Additionally, Hyperparameter tuning was also done for these models. Apart from these models, I also trained many other models but they were not suitable for this particular problem. Moreover, the Neural network-based model was also trained for predicting the service length, However, the r2 score (0.41) obtained was not satisfactory. So, I ignored this model. Since my major contribution is in modelling, my inputs were used in the documentation of these sections. I tried my best to ensure the scalability of the models so that if a new position is opened our chosen model should predict how suitable the candidate is to the organisation. Also, I improved the programs for scalability as well. However, I have to solve the format problem in the salary columns, currently, the code can extract the salary only if it is given in a number format. But, I need to figure out the way that it should be more dynamic. For example, if the salary is given in words then also my code should extract the salary.

As a group, we faced many difficulties while doing this project. The first and most difficult one is that we could not find a dependent variable to feed it into our model, since all the data represented successful candidates. So, the only solution is to consider the problem as unsupervised, but doing so, feature selection on this data might remove relevant columns from the data set. Also, it is challenging to predict the success score as the data given is only for successful candidates. Later, we found that creating a scoring system for this problem will be difficult. Anyway, we tried it, and we developed some solutions with which we were not at all satisfied. So we decided to clarify this in a drop-in session. In the session, I asked some doubts to our module leader and he advised us to use the service length field as the target variable. Also, the quality of the data is not good as we found certain discrepancies, so we had to develop our own justification for the occurrence of some particular values, which may not be correct. For example, we found that in the service length column, there are values like 373 which indicate around 31 years but the company was founded around four years ago, so we assumed that the candidate might have worked in a previous job in different other organisations and we pre-processed accordingly. The values in columns that explain the skills of the candidates are mainly zeroes so this could lead to unexpected results. There are columns in the data which represent the ethnicity and gender of the candidates. So, we dealt carefully with these columns as we need to consider the ethical aspects and neutral nature of our model. The next problem was to figure out how the cookie cutter works for creating the project template. In the earlier modules, we used the R project template for creating the pipeline, but some machine learning modules may not work properly in R and only a few online resources are available for cookie cutter (python). Anyway, one of our team members was allocated to work on the cookie cutter. Another problem we faced was dealing with version control of the code as many people need to work on it. The solution we formed was to separately complete the independent individual task and merge them finally. For example, the data set processed by the data cleaning team will be shared through *Teams* or *Google Drive*) and the feature selection team will work on this data and gives the output to the modelling Team. We ensured that the variables used should not interact with each other. Finally, the code was combined successfully.

While working with my team, I supported the team by participating actively and listening to each person's ideas. I endorsed the idea that I felt was logical and feasible. In several situations, I found that my understanding of certain areas was not exactly correct and the team helped me to improve my knowledge in those areas. I am glad that my team supported me and helped me with certain challenging problems. I think I spent more time researching a particular topic and completed many tasks which are not relevant to this project. If I focused more on a specific topic, I would have completed all the tasks earlier. So that I could have spent some more time on the other two data sets as well. Initially, I thought of using other data sets and building separate Natural language Models to predict the candidate score but I rejected that plan due to time constraints. So, if I get enough time I will consider other data as well and I will produce a much more flexible approach to data modelling. Apart from these, I should have asked my doubts immediately rather than waiting for the drop-in session, Then also I could have saved much more time by completing more tasks rather than spending time on the unsupervised feature selection method. And, I will learn much more about the working of *cookie cutter* in detail so that in future it will not become a problem. These are the main realisation that I experienced from this project.

I consider this a great opportunity as it simulated Teamwork for solving a real-world problem where there are people from different backgrounds. The skills like leadership, listening, communication, problem-solving, and collaboration, were improved by working on this project. I gained more understanding of areas where I thought were correct but not accurate. Also, this new experience taught me that it is important to speak up even though the idea might not be perfect. Because it opens a chance for improving ourselves. Besides that, I learned how to manage time properly in a time-limited environment. Moreover, my technical knowledge of *MySQL* server, *MySQL API*, and linear regression models (random forest regression, linear regression, etc) was improved while working on modelling. And, I think, learning through doing practical work is more effective than any other learning method. Also, I learned the importance of managing disputes happening internally in our team.

As far as I am concerned, the immense support from my team, module leader, and our demonstrator was plentiful. Our demonstrator was really helpful and guided us through many difficult times. For example, there were times when we could not figure out the meaning of certain columns from the data in the business aspect, but they helped us in those areas. Apart from this, our demonstrator clarified many of my doubts while working on my modelling part. Additionally, in the drop-in session, I had a chance to ask our module leader several doubts such as the reason for the discrepancies in the *candidate Id* in the given three datasets, and the data format in the salary column. Their advice was helpful which saved me from spending unnecessary time on irrelevant areas.

Overall, the outcome of this task was useful and provided many opportunities to improve and assess ourselves. The tasks showed how real-world problems can be solved by working as a team. Moreover, it is important to cultivate team skills along with technical skills to attain successful and consistent career growth.