Advanced Analytics

NEDBANK DATA SCIENCE MACHINE LEARNING ENGINEER

Recruitment Problem Pack

Introduction

Hello and welcome to the coding stage of the recruitment process of becoming a machine learning engineer as part of the data science team in the **Advanced Analytics**!

Included in this pack are **three** challenges. In no particular order, you will be tested on: basic machine learning, graphing, feature engineering, optimisation, text manipulation, general problem solving, coding competency. You will also be given the opportunity to conceptual describe how to deploy the model and the considerations to add value to business.

You have **48 hours** to complete as many of them as you can. You are required to choose **at least one** and provide an end to end view of how to solve the problem in order to continue to the next stage of the recruitment process.

We encourage the use of **Python** and **Jupyter Notebooks** to complete the challenges but you are open to use any language to completes the tasks. With deployment, take into consideration that we run a Kubernetes cluster in Azure, security considerations, integration, pipelines and ETL tools, model management and model monitoring. In addition to your code, please add a 2-3 pager conceptual model of the deployment which includes 1 view on ML workflow and a Data Flow diagram describing the integration points of your model in production. Your creativity in presenting the solution in a simple manner will be tested here.

Marks are given for not only completing the challenges successfully but adding your interpretation and thinking at each stage of your process (this would be done best in Markdown in Jupyter, coding comments can also be used).

Good Luck!! 🌞

Note: all scenarios and data provided in these problem are fictional, created only for the purpose of the coding stage of the recruitment process.

Problem 1

The marketing team in Nedbank ran a marketing campaign and recorded who successfully responded to the campaign (where a "success" means they took up the product). They used a "shotgun" approach, meaning they targeted clients at random. Since the campaign required the marketing team to call each client individually, they wanted to save time by only contacting the clients who are "most likely" to take up the product.

The Advanced Analytics (AA) team told them that a model could be developed that could look at their previous campaign and determine which clients are most likely to take up the product. This would allow them to contact clients that have a high likelihood of taking up the product - resulting in a higher hit rate.

Your task is to create this model. You will have been supplied with a cleaned dataset (*bank_marketing_data.csv*) of clients that were targeted in the marketing campaign. The feature 'target' indicates if the call was a success or not.

To do this, you must ingest the data and feed it through three classification models – SVM, Logistic Reg and k-NN – that will predict the outcome. Once you have results for each, choose an appropriate visualisation to communicate your model performances. The accuracy/precision/quality of your model will **not** be taken into consideration (no need to tune hyperparameters).

Problem 2

When unusual amounts are spent on a certain account, the fraud team needs to notify the customer that there is an anomaly and they may potentially be a victim of fraud. Instead of using machine learning, they are trying out a simple outlier rule.

The fraud team want to test out the following rule to notify clients of possible fraud: If a customer on a particular day spends the **same or twice** the median spending amount over a certain number of trailing days, then a warning message is sent to the customer. However, no message is sent until they have a sufficient amount of prior days data.

It is your job to implement this rule. You will be given *d* and a vector of *n* numbers, where *d* is the number of trailing days to check over and *n* is the period of days for expenditure. This vector contains the total amount spent by the customer each day. (see the files "*P2-input1.csv*" and "*P2-input2.csv*")

To test if the rule is working, calculate and display the number of warning messages your model would send the customer. For extra points come up with a good message to send the client with the

Example:

$d = 3$ and $expenditure = [100, 200, 300, 400, 500]$. For the first 3 days, we are collecting data. On day 4, there is a total spend of 400 and we have a median of 200 expenditure over the previous 3 days. Since $400 \geq 2 \times 200$, a notice will be sent. The next day there is a spend of 500, and trailing days give us a median of 300. Since 2×300 is more than 500, no message will be sent. Therefore, over the whole period one warning message is sent.

Note: The median can be found by arranging numbers from smallest to largest. If there are an odd amount of numbers, the middle one is selected. If there are an even amount, the median is defined to be the average of the two middle numbers.

Constraints:

- $1 \leq n \leq 2 \times 10^5$
- $1 \leq d \leq n$
- No spend will be greater than 200

Problem 3

Text fields are not ideal as a joining feature, tables have numerical IDs as their primary/secondary keys for a reason. However, sometimes two data sources from different servers need to be joined and there is no ID linking them. This is the problem the Data Lineage team is struggling with. To solve for this problem, they want to associate a text field in one table with an ID number. Since text is prone to typing errors, white space, synonyms of the same word, and other interesting variations, the field needs to be grouped.

The task is to set up a procedure that will take a text field and group all similar entries to then assign them to an ID. This procedure can be performed on two tables that then can be joined – however, this is not part of the challenge. You have been provided with a file (see "*P3-text.csv*") with 1288 data points. Each entry has a seemingly random set of words, see the following (see Table 1):

text_id	text_field
0	depend bake tree voice labor
1	concert somehow miracle pole
2	concert vital cloud waste

Table 1 - Text field extract

Although "seemingly random", these text fields do hold value. By doing a short analysis, you will find they do overlap (as can be seen in "*concert somehow miracle pole*" and "*concert vital cloud waste*", the word "*concert*" appears in both). Your procedure needs to take advantage of this.

In summary, your task is to: read in the data from the file, transform the text into a suitable format, choose an appropriate clustering/grouping algorithm (you may try multiple and compare them) and, once clustered, assign an ID to each cluster. The above table will then look like something below (see Table 2):

text_id	text_field	cluster_id
0	depend bake tree voice labor	32
1	concert somehow miracle pole	42
2	concert vital cloud waste	14

Table 2 - Cluster ID added

NEDBANK

Note: you are not expected to get the exact IDs listed in this table, and in your example, text_field 1 and 2 may fall into the same cluster – the correctness of your answer is not determined by the value of the "*cluster_id*".

Once you have completed this, you need to visualise/demonstrate your results. Finally, store your text field and associated cluster ID in a file called "*P3-clustered.txt*" and submit it.

Hints:

- Use some form of word vectorizer
- A suitable clustering algorithm/model