Lab Assignment 03

Implementing the End-to-End Data Management and ML Project

This assignment focuses on the end-to-end implementation of machine learning model using Google Cloud Platform services. Mainly Bigquery and Dataprep are used for this project.

. Overview of the data

The dataset is downloaded from UCI machine learning repository. Click here to explore the dataset. This dataset contains details about the student's dropout and academic details from higher education institutions. This supervised dataset has 35 features and 4,424 instances, which are used to predict the students' performance. After analyzing and understanding all variables in the dataset, appropriate cleaning and transformation will be performed to build a classification machine learning model to predict student's performance (whether a student is completed the course or dropout from the course).

Overall process

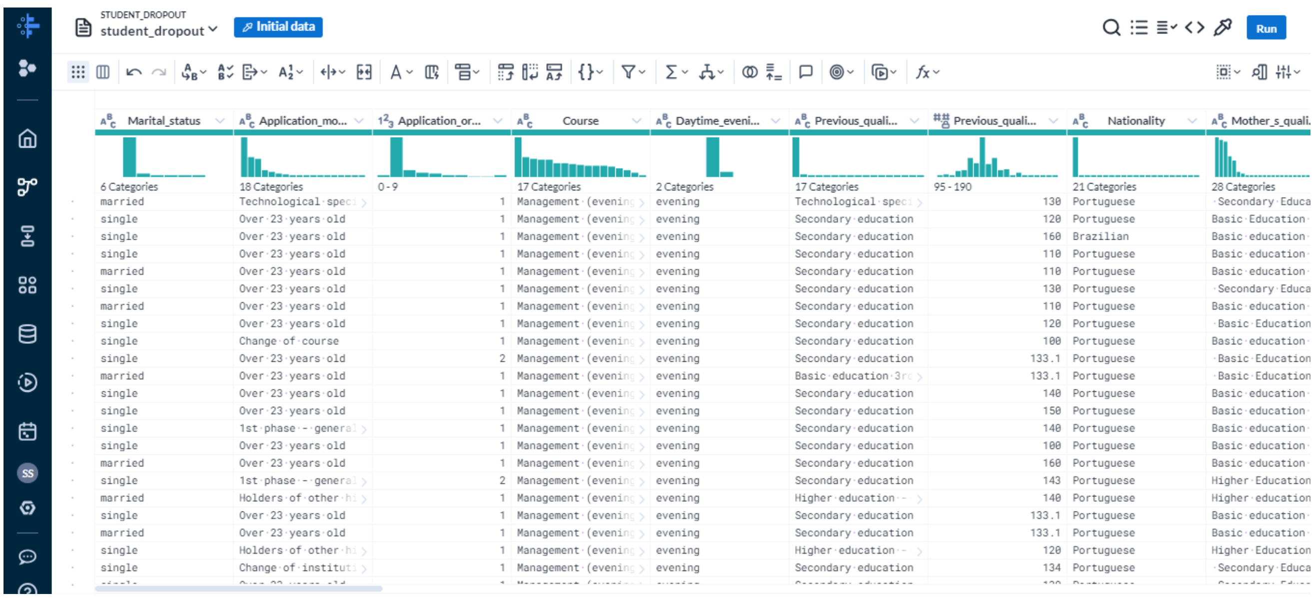
Initially, the dataset is downloaded and stored in Bigquery by creating a database and table. This stored dataset is used in Dataprep to analyze, clean, and transform the necessary variables.
Once the job is completed in Dataprep, the cleaned dataset is stored in Bigquery to perform the model building.

Data preparation using Dataprep

1. In the given dataset, all the categorical variables are in numerical format. So, all these categorical variables are converted into relevant categories to understand their categories and their distribution. For example, the variable "course" has numerical value, but it does not indicate any meaningful information, therefore, the following recipe in Dataprep helps to convert it into its course name.

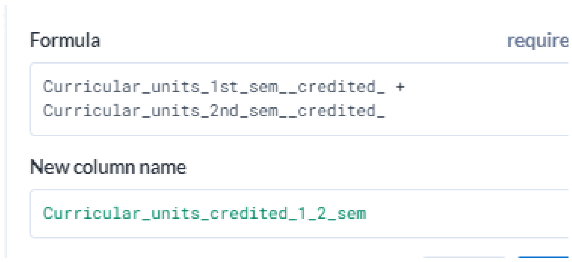
```
8 Set Course to IF(Course == 33, 'Biofuel Production Technologies', IF(Course == 171, 'Animation and Multimedia Design', IF(Course == 8014, 'Social Service (evening attendance)', IF(Course == 9003, 'Agronomy', IF(Course == 9070, 'Communication Design', IF(Course == 9085, 'Veterinary Nursing', IF(Course == 9119, 'Informatics Engineering', IF(Course == 9130, 'Equinculture', IF(Course == 9147, 'Management', IF(Course == 9238, 'Social Service', IF(Course == 9254, 'Tourism', IF(Course == 9500, 'Nursing', IF(Course == 9556, 'Oral Hygiene', IF(Course == 9670, 'Advertising and Marketing Management', IF(Course == 9773, 'Journalism and Communication', IF(Course == 9853, 'Basic Education', IF(Course == 9991, 'Management (evening attendance)', false)))))))))))))))))
```

Similarly, all the categorical variables are converted into their respective categories. The following image shows that the converted variables.



2. Some numerical variables are converted into relevant data types either float or decimal.

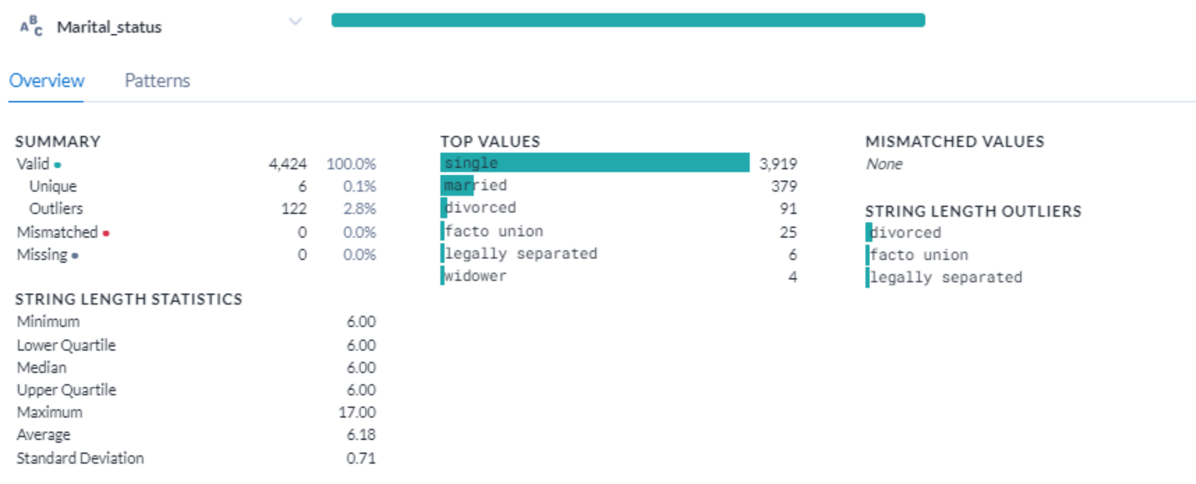
3. The columns contain details of curriculum for first and second semesters are combined to show the total curriculum details for both first and second semesters, For example,



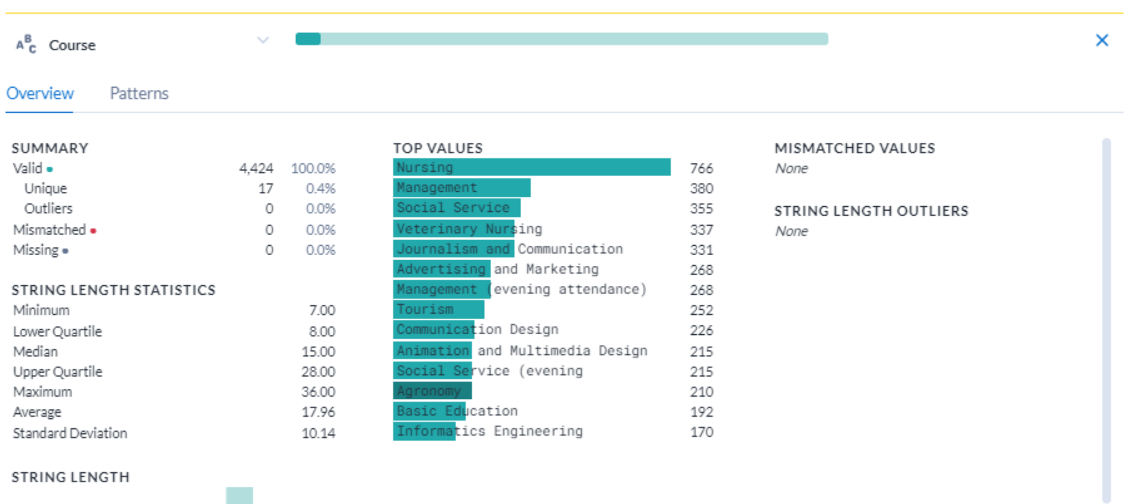
Similarly,
Curricular units 1st sem (credited) + Curricular units 2nd sem (credited) = Curricular_units_credited_1_2_sem
Curricular units 1st sem (enrolled) + Curricular units 2nd sem (enrolled) = Curricular_units_entrolled_1_2_sem
Curricular units 1st sem (evaluations) + Curricular units 2nd sem (evaluations) = Curricular_units_eavaluation_1_2_sem
Curricular units 1st sem (approved) + Curricular units 2nd sem (approved)= Curricular_units_approved_1_2_sem
Curricular units 1st sem (grade) + Curricular units 2nd sem (grade) = Curricular_units_grade_1_2_sem
Curricular units 1st sem (without evaluations) + Curricular units 2nd sem (without evaluations) = Curricular_units_without_evaluations_1_2_sem

4. Label encoding

In the case of one hot encoding, additional variables are created to the dataset that are equal to the categories in each categorical variable. For example, if a variable has 50 categories, then in one hot encoding, it will create 49 additional columns to the dataset. This will increase the size of the dataset and influence the performance of the model. Therefore, only the most frequent variables are selected to perform one hot encoding. For example, the category “single” has higher frequency (more than 88%) in the variable “Marital_status” and other categories are negligible so that it can consider two categories are “single” and “non-single”. Please click here to check the original paper regarding this method of encoding.



This method is applied for all the categorical variable encoding where it has more levels of categories. If multiple categories have high frequencies, then the top 5 categories are considered. For example, the variable "Course" has multiple categories which have reasonable frequencies. Hence the top 5 categories are selected.



While encoding the categorical variables, all the other non-selected categories are considered as "other" and those categories which are similar to drop one of the categories per feature in python one hot encoding. After creating label encoding all the original categorical variables which as categories are deleted.

5. Standardization

Since all the variables are not in a similar scale, standardization is required for some variables. The variables "Previous qualification grade" and "admission grade" are standardized.

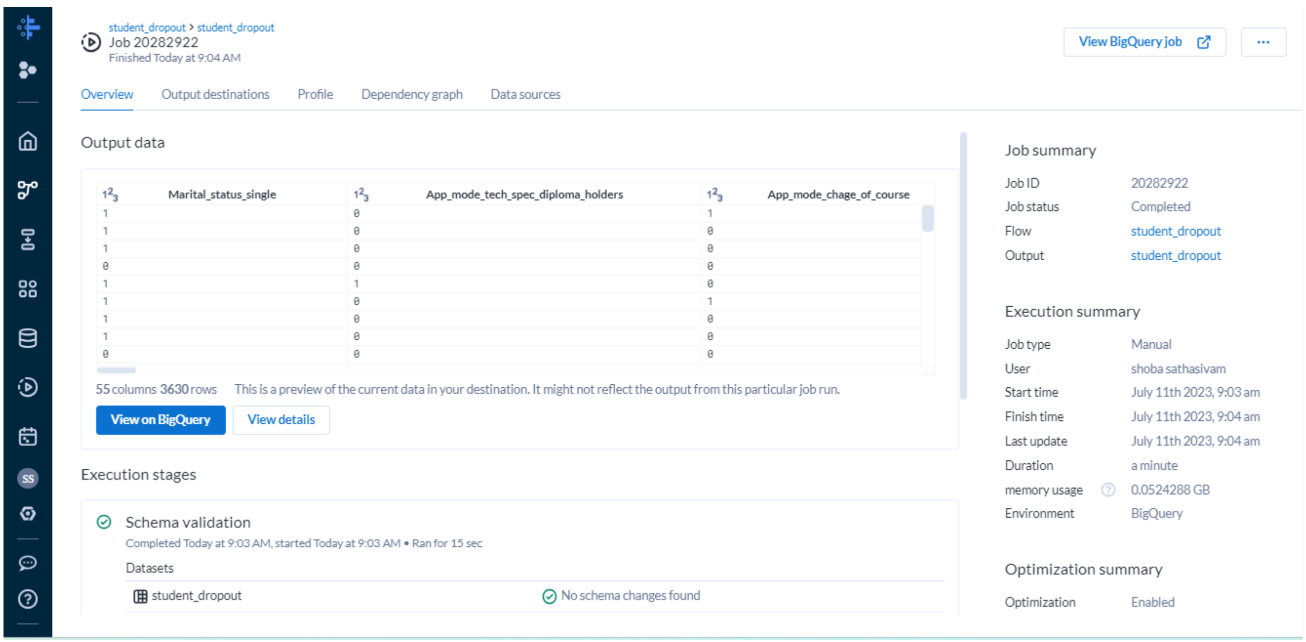
6. No missing values and duplicates in the dataset.

7. There are three categories in response variable such as enrolled, graduate and dropout. However, the problem statement focuses on whether a particular student successfully completed the course or dropped out from the course. Hence, all the enrolled rows are deleted. Therefore, this becomes a binary classification problem.

8. The cleaned and transformed dataset has 55 columns and 3630 rows.

9. To split the dataset for training and testing, an additional column is created “data_split” which has the condition of 80% of the train and 20% of the test. This will help when running queries in Bigquery for model building and evaluation.

10. This cleaned dataset is directed into the created table in Bigquery from Job in Dataprep.



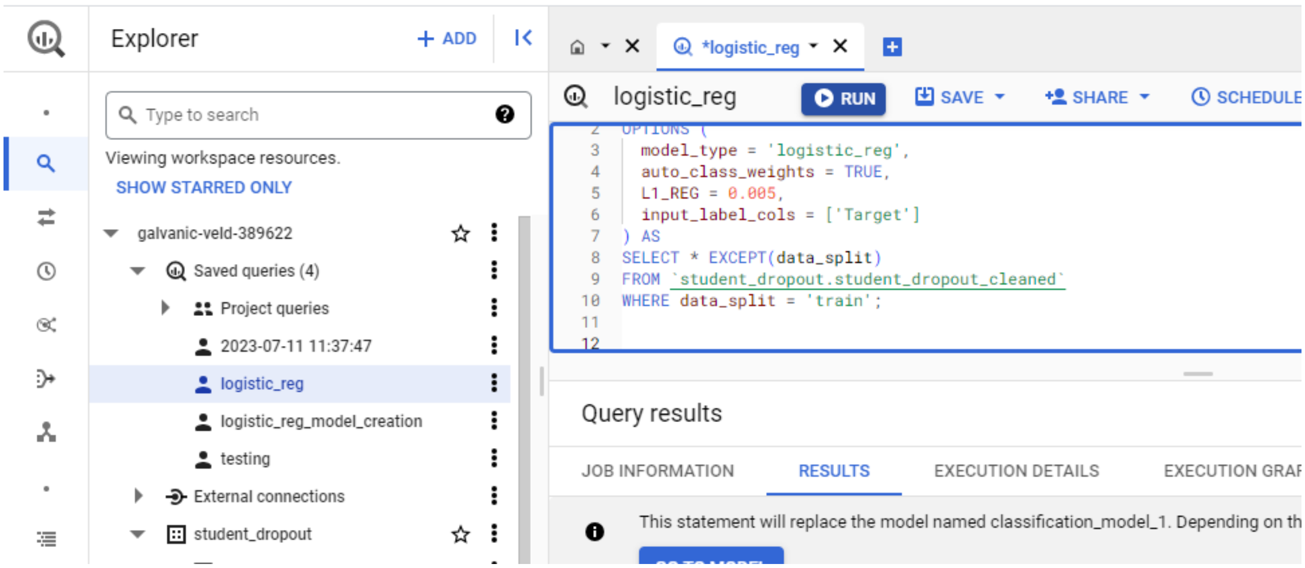
Model building in Bigquery ML

The following SQL query is used to create the basic logistic regression model for this cleaned dataset to predict the binary classification output.

```
CREATE OR REPLACE MODEL `student_dropout.classification_model_1`
OPTIONS (
  model_type = 'logistic_reg',
  auto_class_weights = TRUE,
  L1_REG = 0.005,
  input_label_cols = ['Target']
) AS
```

```sql
SELECT * EXCEPT(data_split)
FROM `student_dropout.student_dropout_cleaned`
WHERE data_split = 'train'
```

Regularization technique called lasso regression (L1_REG = 0.005) is used to avoid the overfitting.



Evaluation of trained model

Aggregate Metrics

Threshold	0.5000
Precision	0.8678
Recall	0.8678
Accuracy	0.8949
F1 score	0.8678
Log loss	0.2908
ROC AUC	0.9430

Score threshold

Positive class threshold	0.5177
Positive class	1
Negative class	0
Precision	0.8782
Recall	0.8636
Accuracy	0.8982
F1 score	0.8708

Confusion matrix

This table shows how often the model classified each label correctly

True label \ Predicted label	1	0
1	86%	14%
0	8%	92%

As per the above result the model is performing well because accuracy is 90% and the precession and recall are more than 85%. Additionally, the confidence threshold is 0.5.

According to the student dropout problem,

Precision = TP/TP+FP – (out of total all students where the algorithm predicted as dropout students, fraction of correctly classified as dropout)

Recall = TP/TP+FN (out of all actual drop out students, fraction of fraction of correctly classified as dropout)

Since FN (student is dropout but classify as graduate) is important in this problem statement, Recall is a measure which decides the model's performance. As per this model recall is 86%, therefore this model is performing well to predict the student's dropout

To get the weights of each variable in the dataset, the following query is run.

```
SELECT * FROM ML.WEIGHTS(MODEL`student_dropout.classification_model_1`);
```

Following is the output of the above query.

Query results

JOB INFORMATION RESULTS JSON EXECUTION DETAILS

Row	processed_input	weight	category_weights.catego
1	Marital_status_single	0.018450868918...	null
2	App_mode_tech_spec_diploma...	-0.58799235574...	null
3	App_mode_chage_of_course	0.091936283738...	null
4	App_mode_over_23_years_old	0.136945466448...	null
5	App_mode_2nd_phase_gen_co...	-0.02230461282...	null
6	App_mode_1st_phase_gen_co...	-0.46600948599...	null

Re

Model evaluation

The following query is run to test the model's performance.

```
SELECT
* FROM
    ML.EVALUATE(MODEL `student_dropout.classification_model_1`,
      (SELECT * EXCEPT(data_split)
      FROM `student_dropout.student_dropout_cleaned`
      WHERE data_split='train')
      );
```

The output of the query is:

precision	recall	accuracy	f1_score	log_loss	roc_auc
0.871662	0.864219	0.895167	0.867925	0.283314	0.945179

Model prediction on test data

```
SELECT
predicted_Target, predicted_Target_probs,Target
FROM
    ML.PREDICT(MODEL `student_dropout.classification_model_1`,
      (SELECT * EXCEPT(data_split)
      FROM `student_dropout.student_dropout_cleaned`
      WHERE data_split='test')
      );
```

Sample of the output is; (Among the total of 692 instances in test data, 626 are correctly classified)

predicted_Target	predicted_Target_probs	Target	Matched
1	{ "predicted_Target_probs": [{ "label": "1", "prob": "0.940783053505192" }, { "label": "0", "prob": "0.05921694649480802" }] }	1	Matched
1	{ "predicted_Target_probs": [{ "label": "1", "prob": "0.99939516738580958"	1	Matched

	}, {
	"label": "0",
	"prob": "0.00060483261419042034"
	}]
	}		

1	{
	"predicted_Target_probs": [{
	"label": "1",
	"prob": "0.99704640907609721"
	}, {
	"label": "0",
	"prob": "0.0029535909239027935"
	}]
	}
		1	Matched

1	{
	"predicted_Target_probs": [{
	"label": "1",
	"prob": "0.9971271636325133"
	}, {
	"label": "0",
	"prob": "0.0028728363674866975"
	}]
	}
		1	Matched

1	{
	"predicted_Target_probs": [{
	"label": "1",
	"prob": "0.99600466916318753"
	}, {
	"label": "0",
	"prob": "0.0039953308368124718"
	}]
	}
		1	Matched

1	{
	"predicted_Target_probs": [{
	"label": "1",
	"prob": "0.99006607651352374"
	}, {
	"label": "0",
	"prob": "0.00993392348647626"
	}]
	}
		1	Matched