BKMS Homework 4

2022-24790 Sungwoo PARK

- For this homework, I chose the Chicago crime dataset provided by the BigQuery public dataset. This data contains the information about each crime case that occurred in Chicago. There are total 7,539,788 samples and the features include the case number, time and space information of each case, and whether the criminal has been arrested or not.
- So here, I tried to make a classification model to predict whether the criminal of each case would be arrested or not with given information of the case.

A. Data Pre-processing

- Here I implemented data pre-processing in two ways: when creating a view for training, and when creating a prediction model.

Pre-processing when creating a view

- The time information of each crime case is given as a datetime format in the original dataset. Therefore, I extracted the year, month, and day information using the function `EXTRACT` in SQL.
- Also, I created a feature called `annotation` to annotate which rows are going to be used for training, evaluation(or validation), and prediction(or test). Specifically, I used the modulo operation for data split - divide the key of each case by 10, then set the samples whose remainder is less than 8 as the training set, set the ones whose remainder equals to 8 as the evaluation, and the rest as the prediction set.
- Finally, I dropped the missing values using a `WHERE` condition.
- The full query I used to create the view for training is shown below.

```
CREATE OR REPLACE VIEW `chicago_crime.crime_view` AS
SELECT
EXTRACT(year from date) AS year,
EXTRACT(month from date) AS month,
EXTRACT(day from date) AS day,
block, iucr, primary_type, arrest, domestic,
district, ward, community_area,
CASE
  WHEN MOD(unique_key, 10) < 8 THEN "training"
  WHEN MOD(unique_key, 10) = 8 THEN "evaluation"
  ELSE "prediction"
END AS annotation
FROM `bigquery-public-data.chicago_crime.crime`
WHERE district IS NOT NULL
```

Pre-processing when creating a prediction model

- There is a feature pre-processing method provided by BigQuery, called `TRANSFORM`. Variables included in the `TRANSFORM` clause are automatically pre-processed following the pre-defined rules. Specifically, numerical variables are standardized(zero-mean and unit standard deviation) and others including boolean, text, date variables are one-hot encoded.
- However, the numerical variables in the data I used are more like categorical labels; they have no arithmetic meaning. So I rather used the `ML.BUCKETIZE` function, which is provided by the BigQuery as well, to make those variables into categorical features.
- The query I used is shown in the box below.

```
TRANSFORM(
  block, iucr, primary_type, arrest, domestic,
  ML.BUCKETIZE(year, [2001, 2002, 2003, 2004, 2005, 2006, 2007, 2008, 2009, 2010, 2011,
  2012, 2013, 2014, 2015, 2016, 2017, 2018, 2019, 2020, 2021, 2022]) AS year_bucket,
  ML.BUCKETIZE(month, [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12]) AS month_bucket,
```

```
  ML.BUCKETIZE(district, [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12,
  14, 15, 16, 17, 18, 19, 20, 21, 22, 24, 25, 31]) AS district_bucket,
  ML.BUCKETIZE(ward, [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16,
  17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34,
  35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49]) AS ward_bucket,
  ML.BUCKETIZE(community_area, [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16,
  17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34,
  35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49]) AS comm_bucket
)
```

B. BigQuery ML Model

- BigQuery ML model provides several candidates for classification: LOGISTIC_REG, BOOSTED_TREE_CLASSIFIER, DNN_CLASSIFIER, DNN_LINEAR_COMBINED_CLASSIFIER
  , AUTOML_CLASSIFIER. Among these, I chose to train a logistic regression model and a boosted tree model and compare the training result.
- Since I have already annotated which data to use for training, validation, and prediction, I gave an option `data_split_method` as `“NO_SPLIT”`. The entire query I used is shown in the box below.

```
-- logistic regression model
CREATE OR REPLACE MODEL `chicago_crime.crime_logistic1`
TRANSFORM(
  block, iucr, primary_type, arrest, domestic,
  ML.BUCKETIZE(year, [2001, 2002, 2003, 2004, 2005, 2006, 2007, 2008, 2009, 2010, 2011,
  2012, 2013, 2014, 2015, 2016, 2017, 2018, 2019, 2020, 2021, 2022]) AS year_bucket,
  ML.BUCKETIZE(month, [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12]) AS month_bucket,
  ML.BUCKETIZE(district, [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12,
  14, 15, 16, 17, 18, 19, 20, 21, 22, 24, 25, 31]) AS district_bucket,
  ML.BUCKETIZE(ward, [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16,
  17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34,
  35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49]) AS ward_bucket,
  ML.BUCKETIZE(community_area, [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16,
  17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34,
  35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49]) AS comm_bucket
)
OPTIONS (
  model_type = "LOGISTIC_REG",
  auto_class_weights=TRUE,
  data_split_method="NO_SPLIT",
  input_label_cols=["arrest"]
) AS
SELECT *
FROM `chicago_crime.crime_view`
WHERE annotation="training"

-- boosting model
CREATE OR REPLACE MODEL `chicago_crime.crime_tree1`
TRANSFORM(
  block, iucr, primary_type, arrest, domestic,
  ML.BUCKETIZE(year, [2001, 2002, 2003, 2004, 2005, 2006, 2007, 2008, 2009, 2010, 2011,
  2012, 2013, 2014, 2015, 2016, 2017, 2018, 2019, 2020, 2021, 2022]) AS year_bucket,
  ML.BUCKETIZE(month, [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12]) AS month_bucket,
  ML.BUCKETIZE(district, [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12,
  14, 15, 16, 17, 18, 19, 20, 21, 22, 24, 25, 31]) AS district_bucket,
  ML.BUCKETIZE(ward, [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16,
  17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34,
  35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49]) AS ward_bucket,
  ML.BUCKETIZE(community_area, [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16,
  17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34,
  35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49]) AS comm_bucket
)
OPTIONS (
  model_type = "BOOSTED_TREE_CLASSIFIER",
  auto_class_weights=TRUE,
  data_split_method="NO_SPLIT",
  input_label_cols=["arrest"]
) AS
SELECT *
FROM `chicago_crime.crime_view`
WHERE annotation="training"
```

- The evaluation of each model was implemented via the query shown in the code box below.

```
-- logisitc model
SELECT * FROM
ML.EVALUATE(MODEL `chicago_crime.crime_logistic1`,
(
  SELECT *
  FROM `chicago_crime.crime_view`
  WHERE annotation="evaluation"
 )
)

-- boosting model
SELECT * FROM
ML.EVALUATE(MODEL `chicago_crime.crime_tree1`,
(
  SELECT *
  FROM `chicago_crime.crime_view`
  WHERE annotation="evaluation"
 )
)
```

- Finally, the prediction was implmented via the query below.

```
-- logistic model
SELECT * FROM
ML.PREDICT(MODEL `chicago_crime.crime_logistic1`, (
   SELECT * FROM `chicago_crime.crime_view`
   WHERE annotation="prediction"
 )
)

-- boosting model
SELECT * FROM
ML.PREDICT(MODEL `chicago_crime.crime_tree1`, (
   SELECT * FROM `chicago_crime.crime_view`
   WHERE annotation="prediction"
 )
)
```

C. Analysis

Reasons for pre-processing

- I thought that the date information itself is too detailed and may induce an overfitting problem, so I split it into year, month, and day, then aggregated each variable into categories.
- For distirct, ward, and community area variables, although they have a data type of integer, while they have no arithmetic meaning: they are closer to categories indicating the place where each crime is committed.
  - However, the BigQuery ML model only uses the variables included in the `TRANSFORM` clause if it is used. This implies that even the categorical variables are standardized only due to the fact that they are integer. To avoid this, I used `ML.BUCKETIZE` to convert them into categories.
  - Year, month, and day were bucketized for the same reason.
- The other variables are one-hot encoded by the `TRANSFORM` clause.

Reasons for selecting model

- I selected the logisitic regression and the boosted tree model to compare the evaluation result. The reason why I selected those models is that I know how they work, since I learned them in another course. I have just little idea about the deep neural network model, I could not use it.

Model Evaluation & Prediction Result

- The evaluation query yields result with 6 different metrics: prediction, recall, accuracy, F1-score, log loss, and ROC-AUC score. All of the metrics are used in binary classification. For details,
  - Accuracy is the ratio of the number of samples whose predicted value is equal to its true value.

- Precision is the ratio of the true positive(the true label and the predicted label are both 1) over the cases predicted to be positive, while recall is the ratio of the true positive over the case truly positive.
- F1-score and ROC-AUC score are calculated from the precision and recall, where F1-score is the harmonic mean of the precision and recall, and ROC-AUC computes the area under the ROC curve, which is plotted by the recall and false positive rate.
- Finally, the log loss shows how much the predicted values are different from the true values.

- At the default threshold, 0.5, the metrics from two models are shown in the table below.

	Precision	Recall	Accuracy	F1 score	Log loss	ROC-AUC
Logistic Regression	0.7686	0.7180	0.8676	0.7425	0.3522	0.9026
Boosted Tree	0.7770	0.7022	0.8672	0.7377	0.3753	0.8966

- Finally, after prediction, I retrieved the most important feature, in terms of feature importance, by using `EXPLAIN_MODEL` command. However an error occurred when I tried this command when using the tree model.
- The part of the result is shown in the figure below. As we can see, the type of crime and the iucr, which stands for Illinois Uniform Crime Reporting, the official classification code for crime reporting in Illinois, are top-two most important features.

Row	predicted_arrest	probability	top_feature_attributions
1	false	0.78820781008745522	[{ "feature": "primary_type", "attribution": "-0.43617237644083895" }]
2	true	0.952371125050023	[{ "feature": "primary_type", "attribution": "1.5888143659977483" }]
3	false	0.54428074379745106	[{ "feature": "iucr", "attribution": "0.49517708238369473" }]
4	false	0.61181261353979577	[{ "feature": "iucr", "attribution": "0.43233064849411063" }]
5	false	0.76449945546361231	[{ "feature": "block", "attribution": "-0.61183224068814923" }]
6	true	0.65922071652195968	[{ "feature": "iucr", "attribution": "0.675457307614233" }]
7	false	0.8857282331867482	[{ "feature": "iucr", "attribution": "-1.1350495345032348" }]
8	true	0.9316888786978228	[{ "feature": "iucr", "attribution": "1.7455827363103047" }]
9	false	0.70902881331702106	[{ "feature": "primary_type", "attribution": "-0.60952444137398576" }]
10	true	0.57216494896500514	[{ "feature": "iucr", "attribution": "2.30902277910608" }]

The most important feature computed from the logistic model - part of example

How to improve the result

- Since the district, ward, and community area variables are about the details of the place where the crime is committed, I thought that there could exist some collinearity with the block variable. Therefore, I excluded them from the model, and then implemented the training again. The evaluation result is shown in the table below.

	Precision	Recall	Accuracy	F1 score	Log loss	ROC-AUC
Logistic Regression	0.8399	0.6816	0.8808	0.7525	0.3150	0.9037
Boosted Tree	0.7631	0.7104	0.8643	0.7358	0.3715	0.8990

- After dropping the variables, we can see that both models show better performance, in terms of log loss especially. But still, the performance of the logistic model is better than that of the boosted tree model.
- Furthermore, I heard that the hyper-parameter tuning can be implemented by BigQuery as well. So I think I can improve the performance of the model by tuning the hyper-parameters for each model.