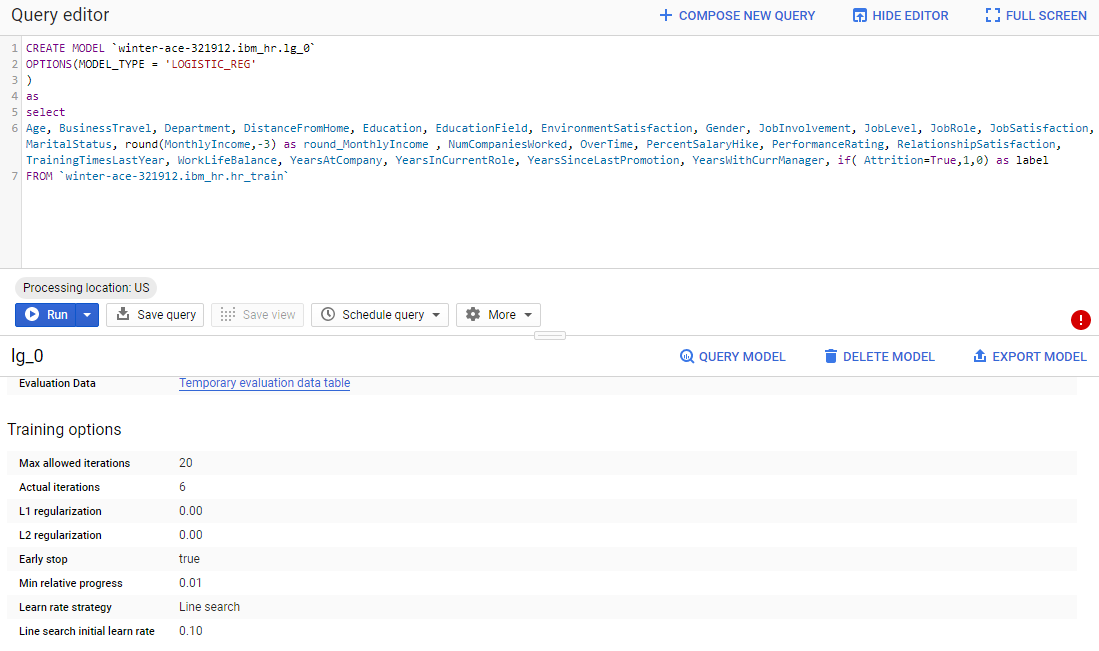
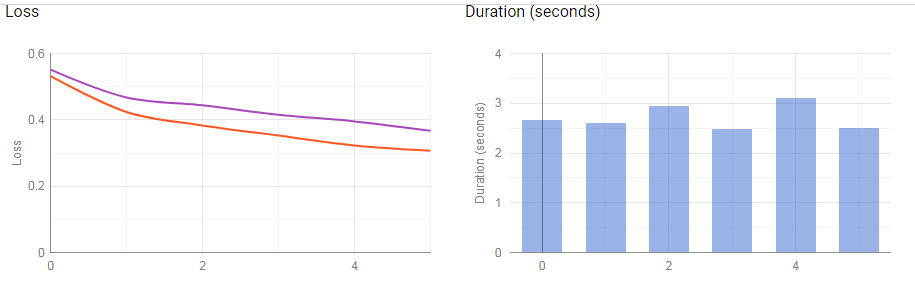
# Задание 2. Модуль 3. Нуриева

## Логистическая регрессия

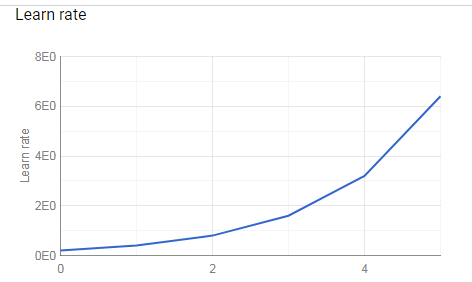
### Модель 1



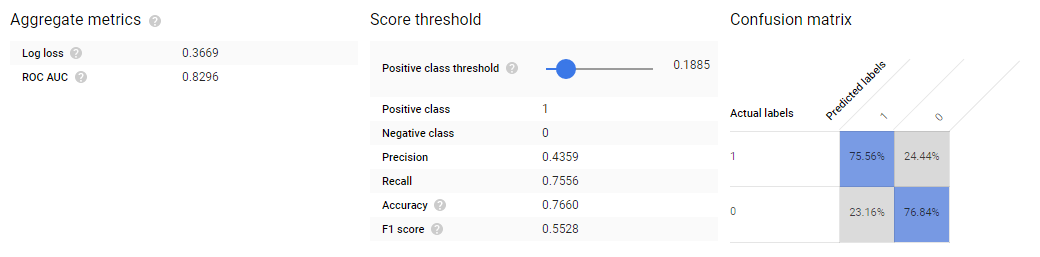
Создаем модель обучения на основе логистической регрессии с базовыми параметрами. Входящими параметрами возьмем почти все столбцы (за исключением тех, в которых значения одинаковые для всех строк)



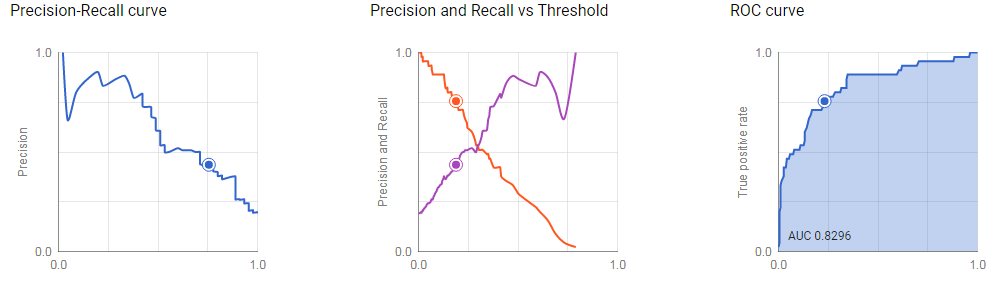
Показатели модели: Функция Loss (Оранжевый график – данные, получение в результате обучения, фиолетовый – в результате оценки), время обучения модели на каждую итерацию.



Рейтинг обучения



Оценка метрик модели: порог отсечки, матрица ошибок.

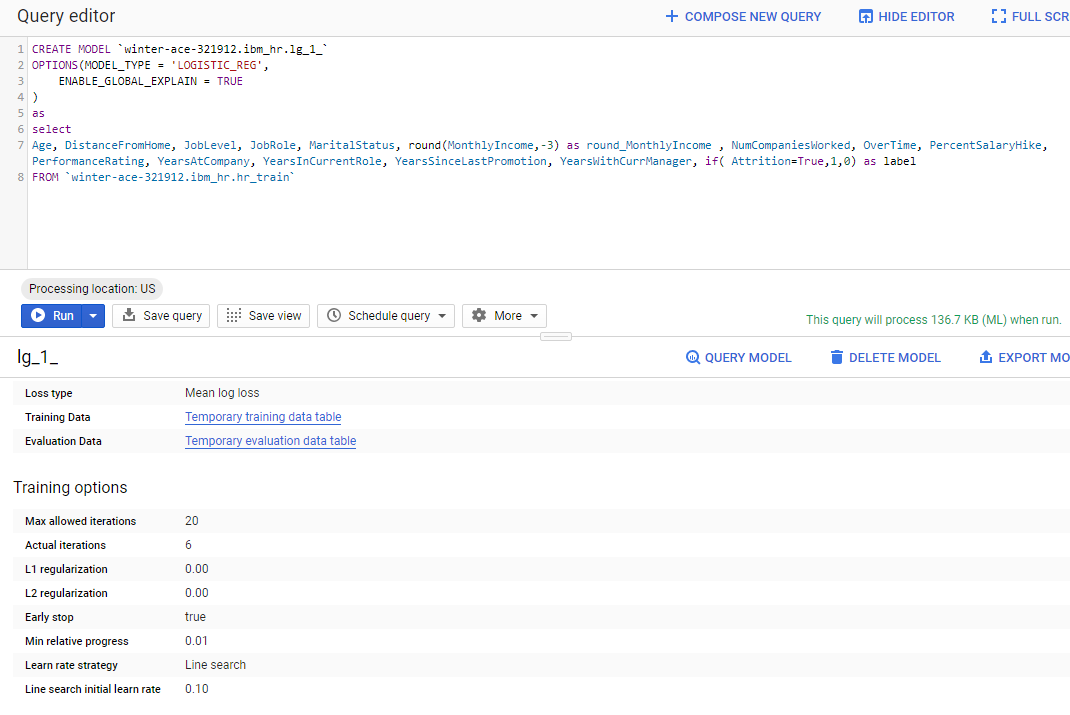


Оценка метрик модели: кривые точности-полноты; точности, полноты и порога отсечения и рок-кривая.

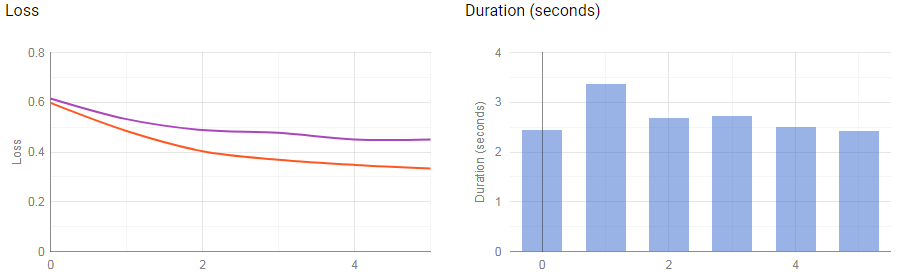


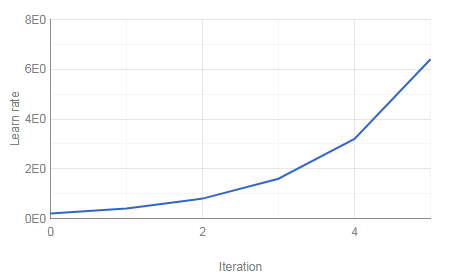
Метрики первой модели

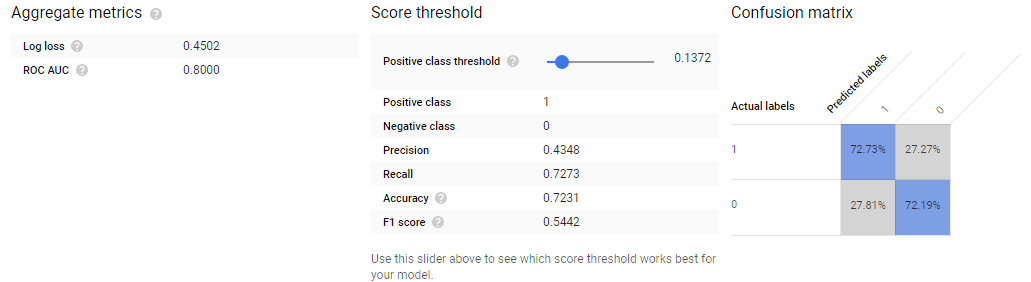
### Модель 2

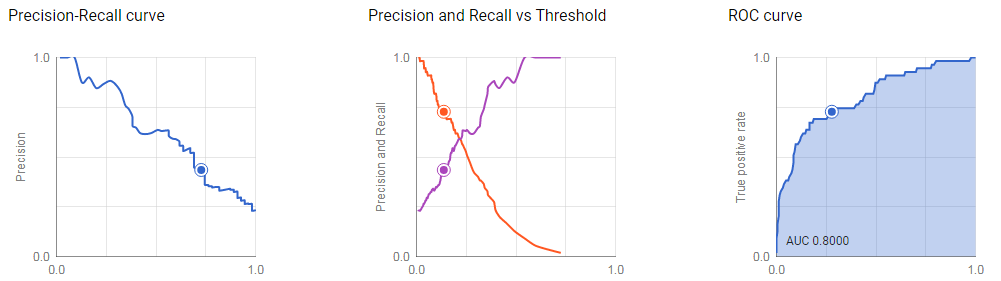


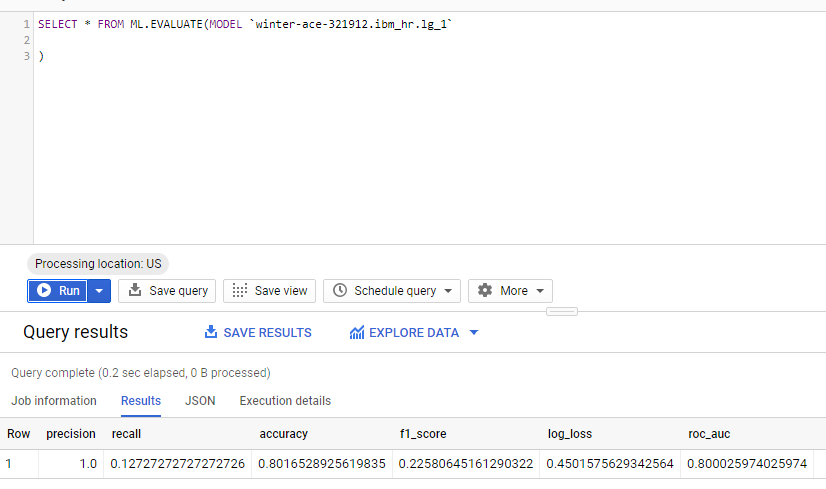
Вторую модель создаем с меньшим количеством входных параметров и включим опцию «ENABLE\_GLOBAL\_EXPLAIN», которая позволит посмотреть «объяснение» модели.

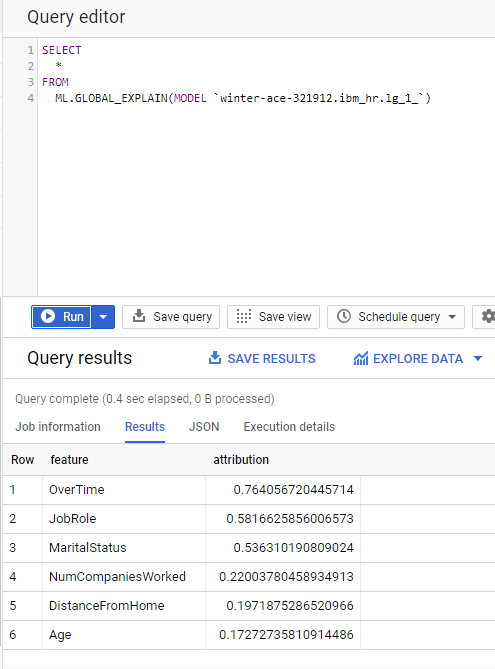






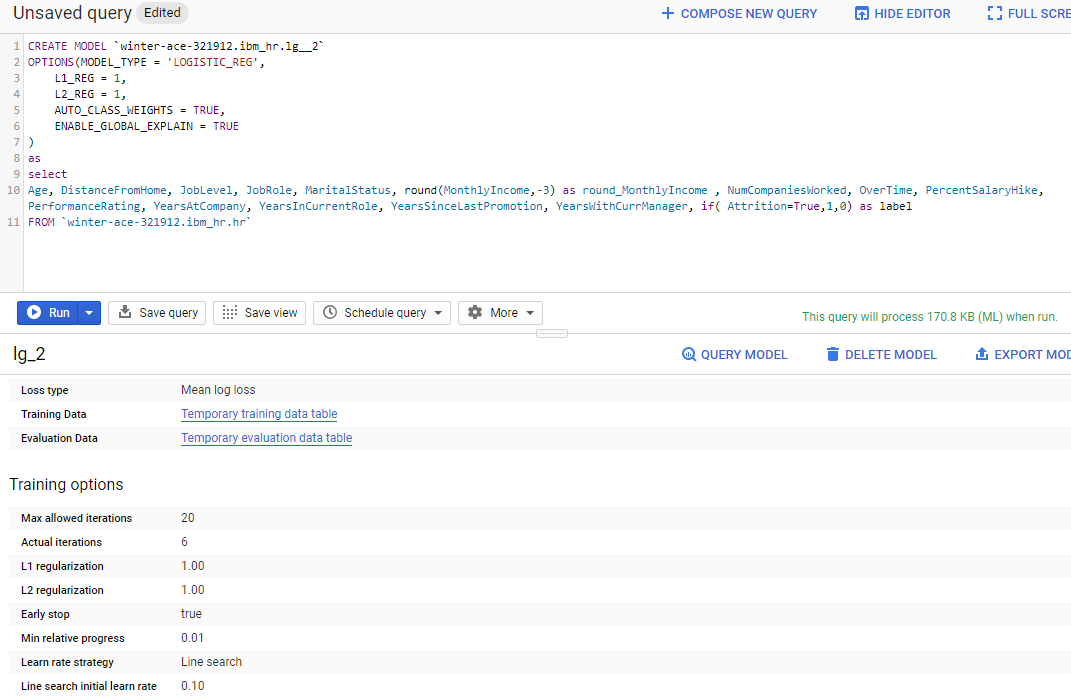




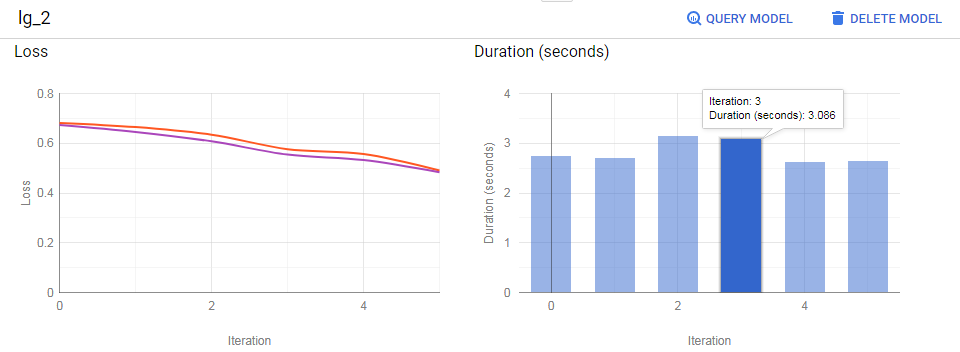


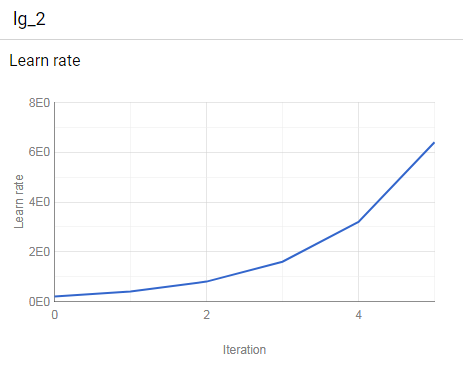
Сделаем запрос «объяснения» модели и увидим, какие параметры модель посчитала важными: переработки; должность; семейное положение; количество компаний, в которых работал; расстояние до дома; возраст.

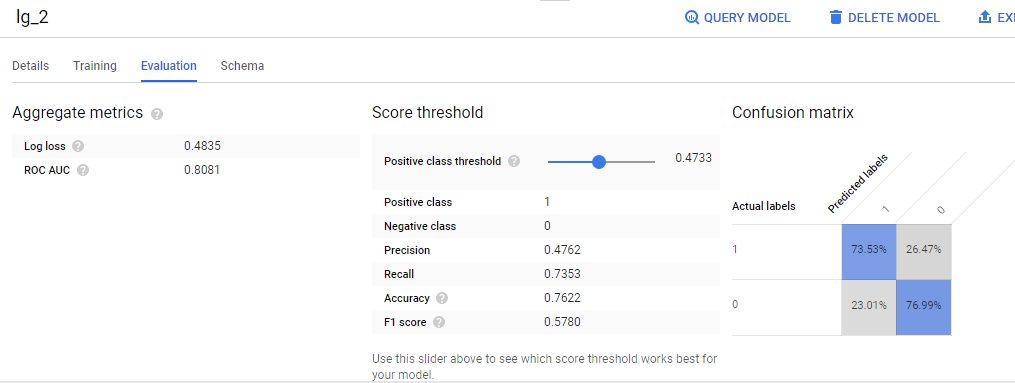
### Модель 3

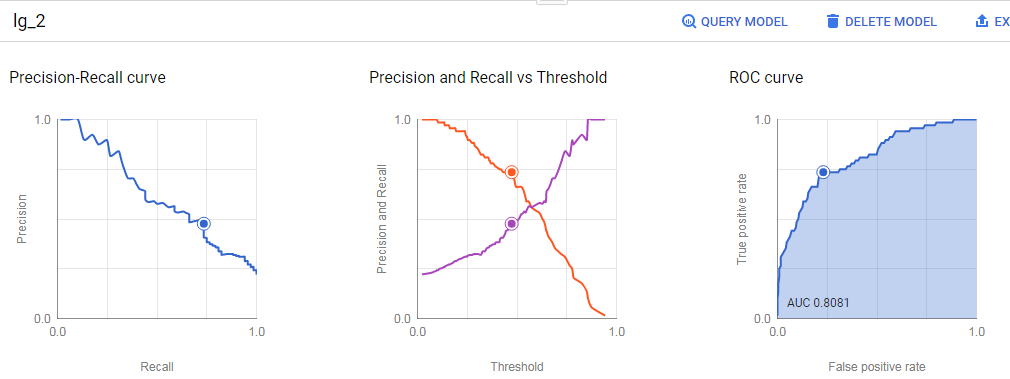


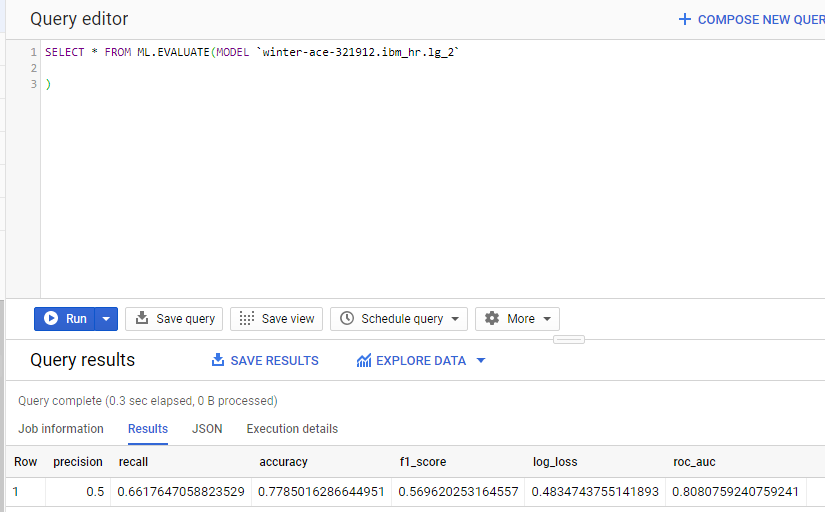
Построим третью модель, включив больше опций.



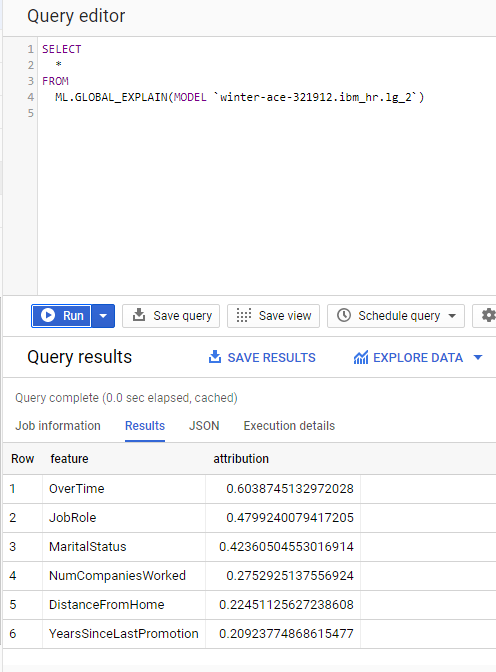




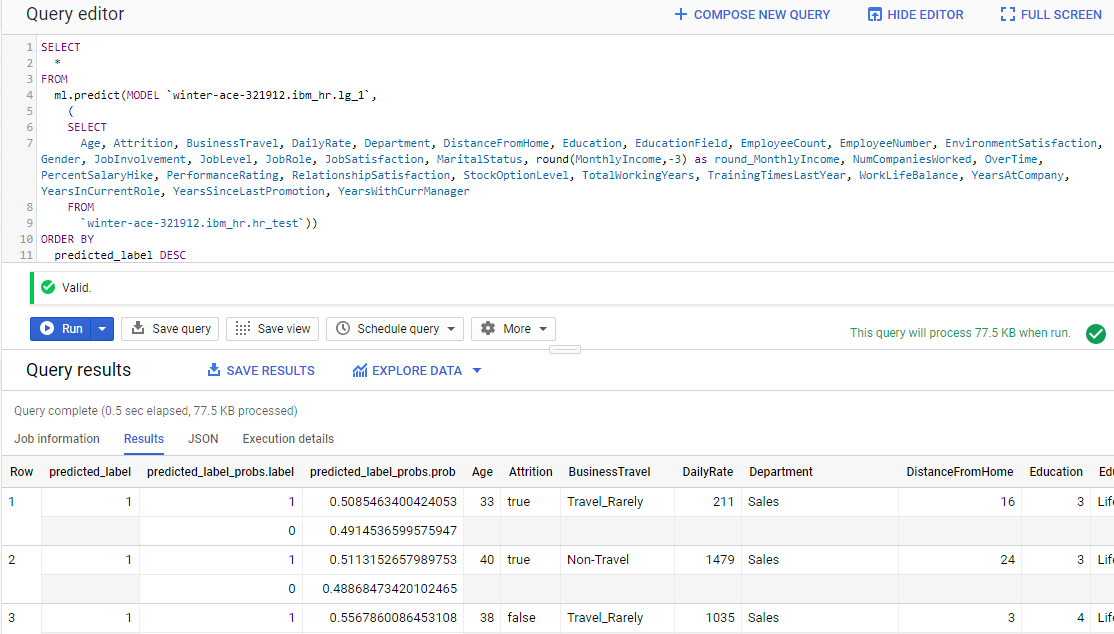




Значения метрик оказались меньше, чем у первой модели.



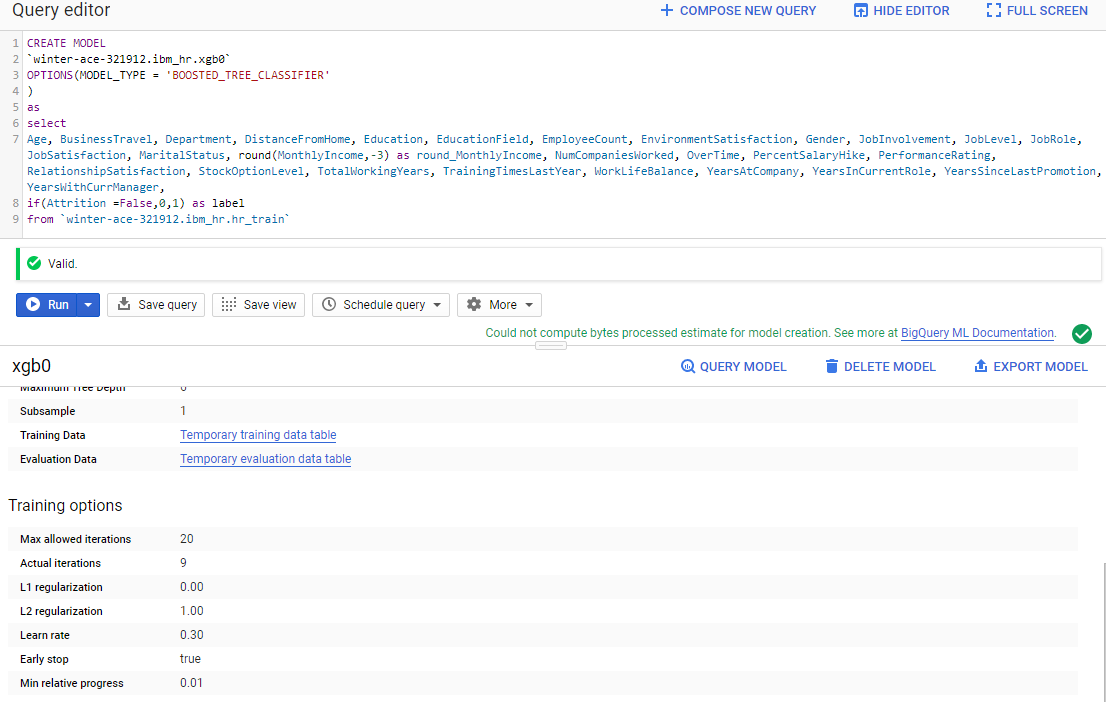
Модель посчитала важными параметры, почти такие же как во второй модели, кроме последнего: годы с последнего повышения вместо возраста.



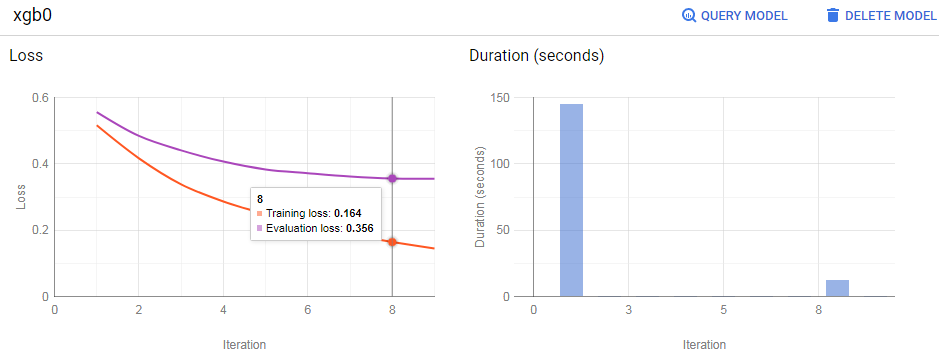
Делаем прогноз на тестовых данных.

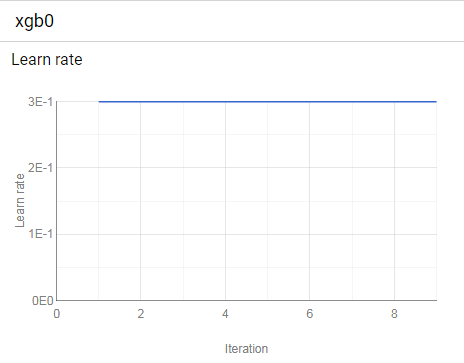
## XGBoost

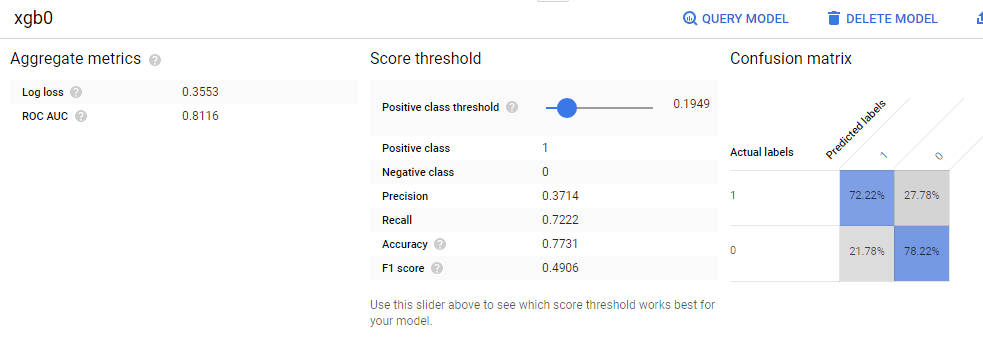
### Модель 1

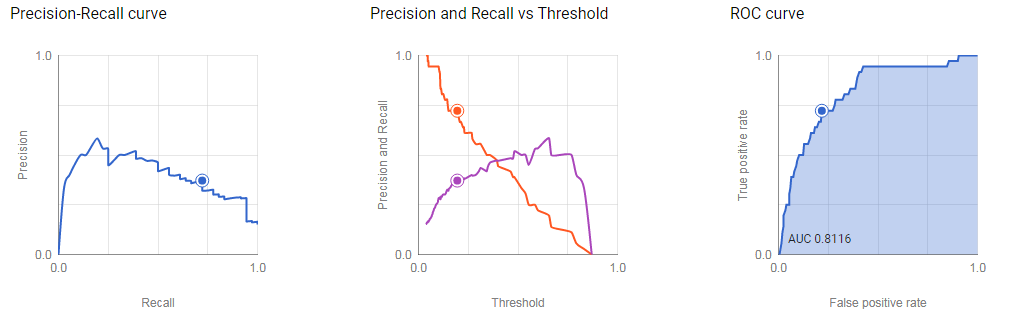


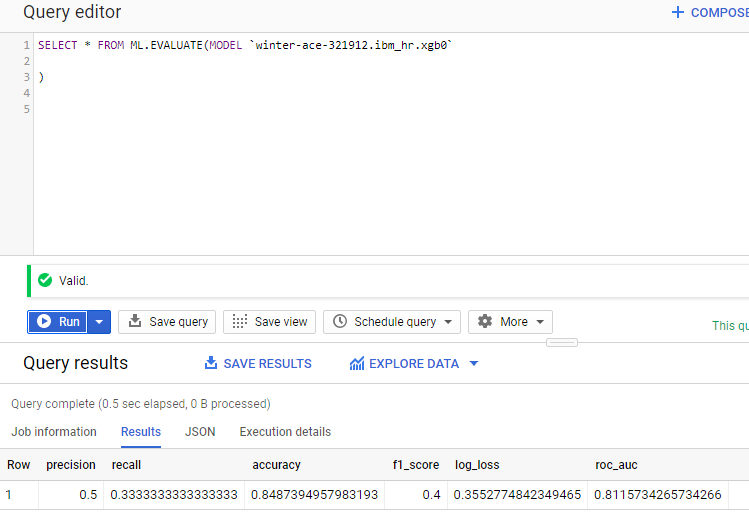
Создаем модель обучения на основе XGBOOST с базовыми параметрами. Входящими параметрами возьмем почти все столбцы (за исключением тех, в которых значения одинаковые для всех строк)

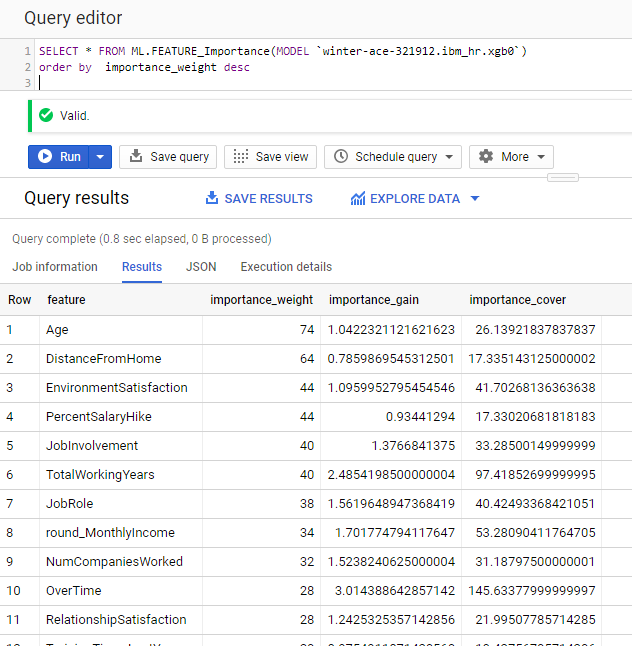




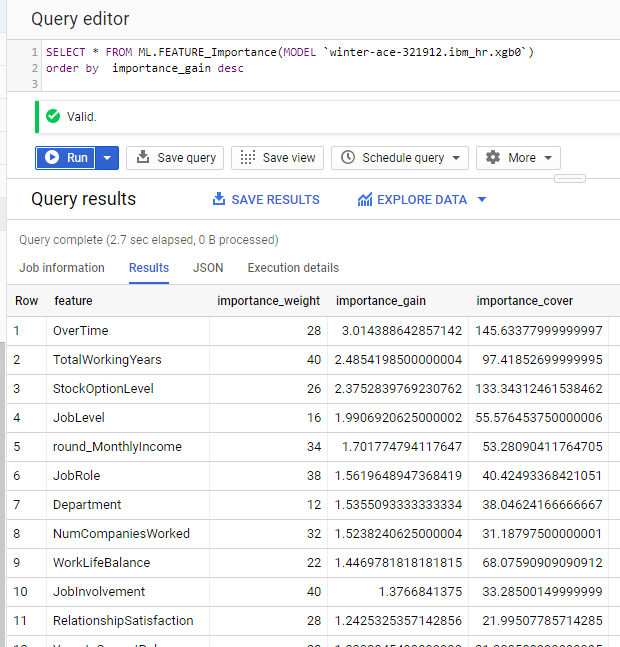




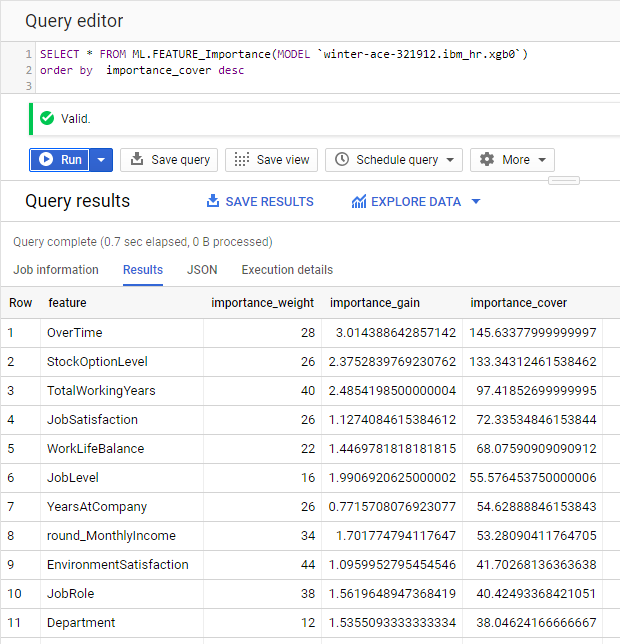




Смотрим параметры, которые модель посчитала важными. Соритуем в порядке убывания значения “Вес важности” - количество раз, когда функция используется для разделения данных по всем деревьям.

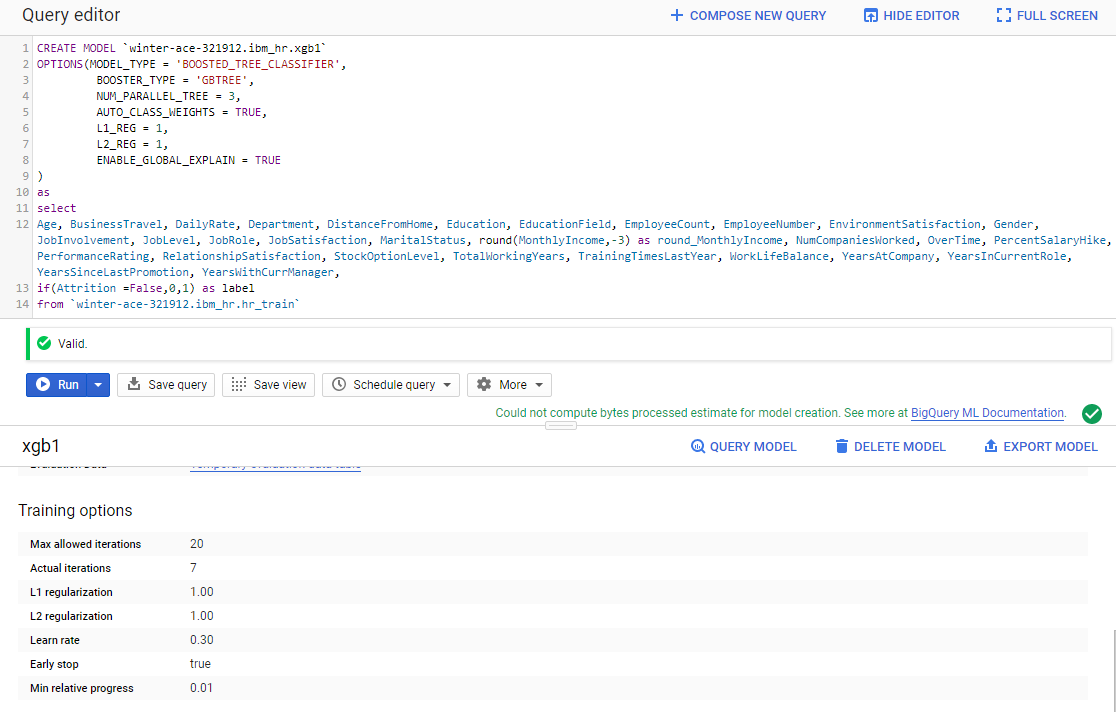


А здесь сортировка по значению “Важность усиления” - средний коэффициент усиления по всем разделам, в которых используется функция.

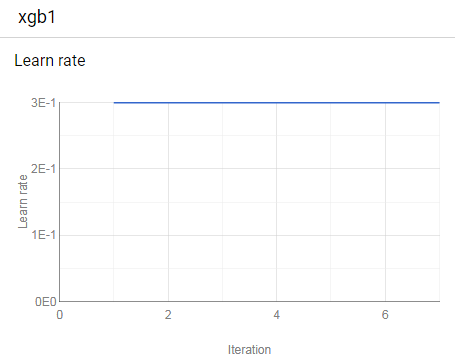
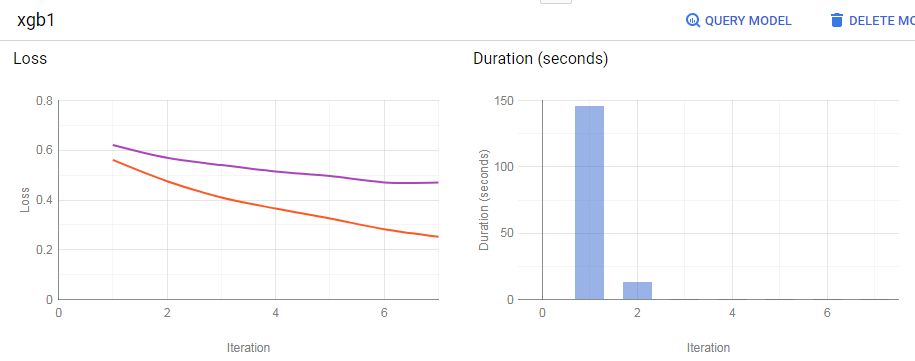


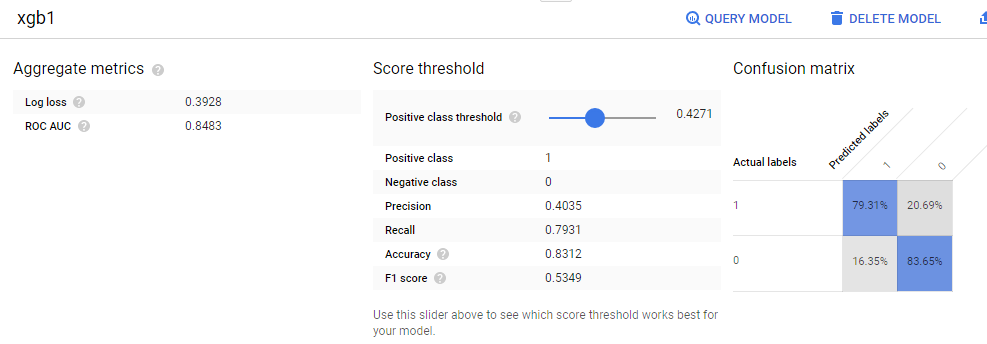
А здесь сортировка по значению “Важность покрытия”- среднее покрытие по всем разделам, в которых используется объект.

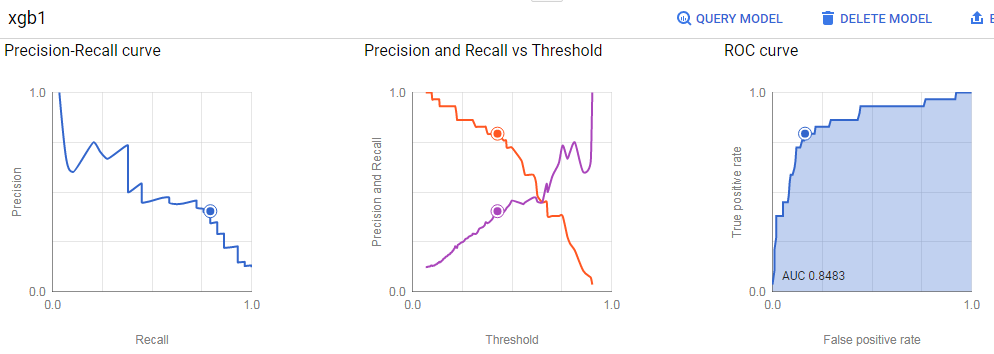
### Модель 2

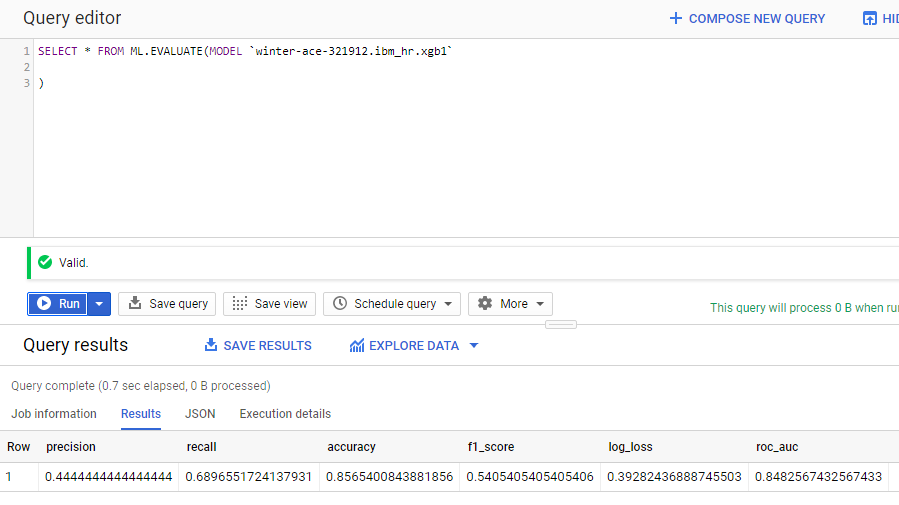


Вторую модель создадим, включив больше опций.

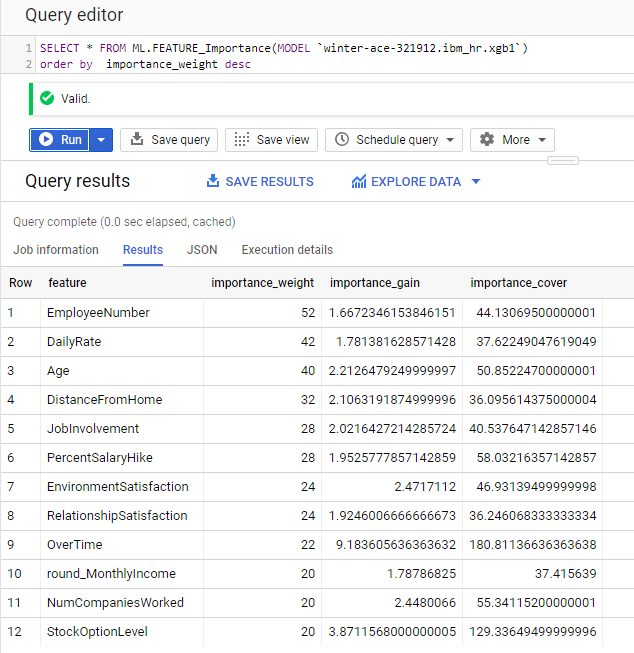




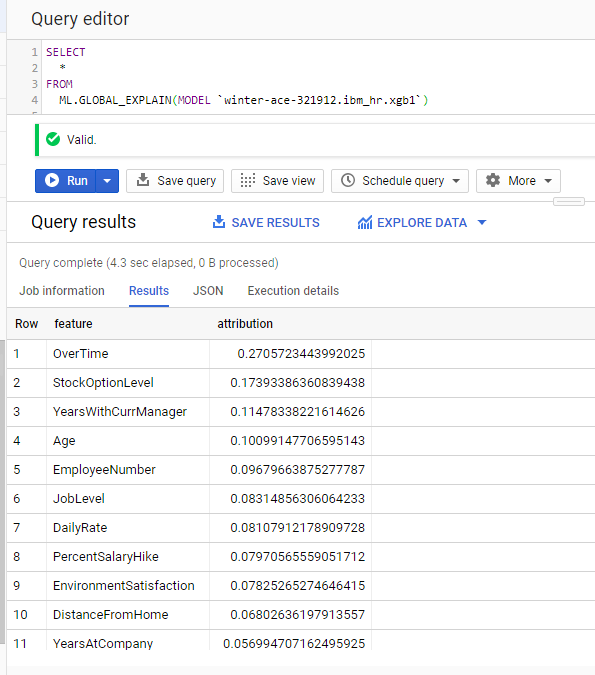




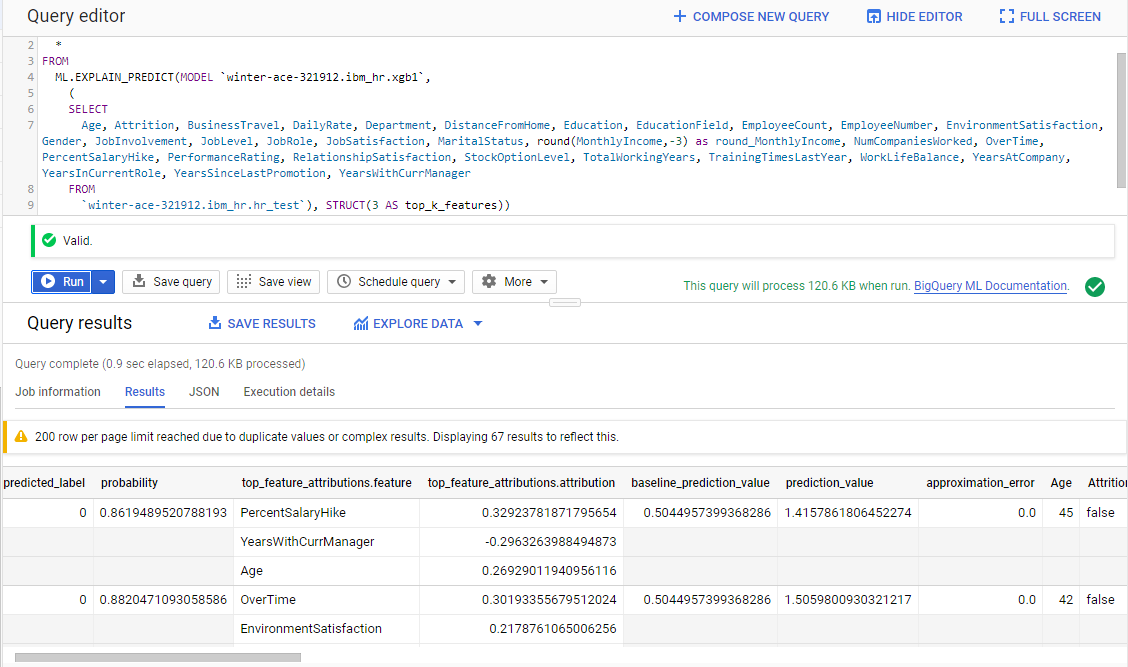
Метрики оказались в среднем чуть лучше, чем у первой модели XGBOOST и всех моделях логистической регрессии.



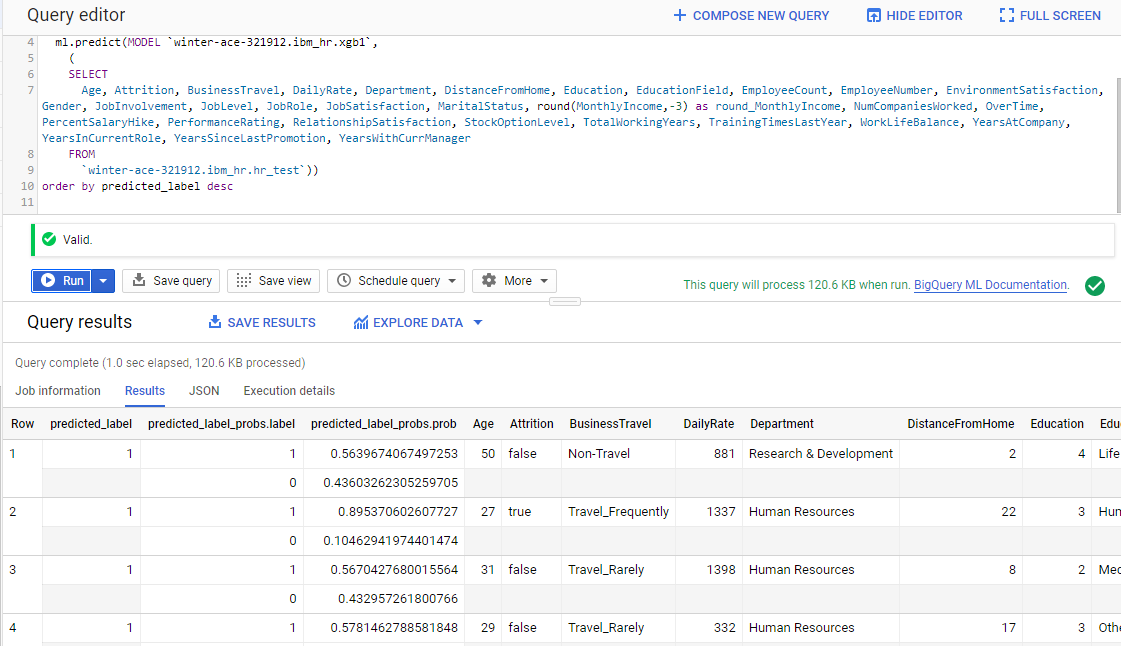
Важные параметры по версии модели.



Но функция “Объяснения” модели показала немного другие важные параметры.



Можно сделать запрос, формирующий прогноз и выводящий дополнительные значения, объясняющие модель.



А можно вызвать функцию, формирующую только прогноз.

## Выводы

Метрики, полученные в результате машинного обучение в BigQuery:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Метрики | Логистическая регрессия 1 | Логистическая регрессия 2 | Логистическая регрессия 3 | XGBOOST 1 | XGBOOST 2 |
| Accuracy | 0.855 | 0.802 | 0.779 | 0.849 | 0.857 |
| Precision | 0.867 | 1.0 | 0.5 | 0.5 | 0.444 |
| Recall | 0.289 | 0.127 | 0.662 | 0.333 | 0.6896 |
| roc\_auc | 0.8296 | 0.8 | 0.808 | 0.812 | 0.848 |
| f1\_score | 0.433 | 0.226 | 0.57 | 0.4 | 0.541 |
| log\_loss | 0.367 | 0.45 | 0.483 | 0.355 | 0.393 |

Метрики, полученные в результате машинного обучение в Colab с помощью Python:

| **Metrics** | **Logistic**  **Regression** | **Logistic**  **Regression**  **tuned** | **Logistic**  **Regression**  **final** | **Gradient**  **Boosting**  **Classifier** | **Gradient Boosting Classifier tuned** | **Gradient Boosting Classifier final** |
| --- | --- | --- | --- | --- | --- | --- |
| Accuracy | 0.8697 | 0.8697 | 0.9122 | 0.8414 | 0.8499 | 1.0 |
| Precision | 0.7647 | 0.8214 | 0.9714 | 0.6667 | 0.7619 | 1.0 |
| Recall | 0.4062 | 0.3594 | 0.5312 | 0.25 | 0.25 | 1.0 |
| Roc\_auc | 0.6893 | 0.671 | 0.7639 | 0.6112 | 0.6163 | 1.0 |

Лучше всего метрики у модели, полученной в Python, созданной после двух видов улучшений.