МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«Национальный исследовательский ядерный университет «МИФИ»
Обнинский институт атомной энергетики –
филиал федерального государственного автономного образовательного учреждения высшего образования «Национальный исследовательский ядерный университет «МИФИ»
(ИАТЭ НИЯУ МИФИ)

Отделение интеллектуальных кибернетических систем

ЛАБОРАТОРНАЯ РАБОТА №4
«Машинное обучение »
по дисциплине
«Большие данные»

Выполнил студент 1 курса
группы ИВТ-М20
Лискунов Р. Г.

Проверил:
кандидат технических наук
Грицюк С. В.

Обнинск, 2020

Цель работы

Определить простую задачу машинного обучения и решить ее. Точность на испытательном наборе должна быть не менее 60%.

Краткая теория

Apache Spark MLlib используется для создания приложения машинного обучения. Приложение выполняет прогнозный анализ на открытом наборе данных. MLlib — это основная библиотека Spark, которая предоставляет множество служебных программ, полезных для задач машинного обучения, таких как:

1. Классификация;
2. Регрессия;
3. Кластеризация;
4. Моделирование сингулярного разложения и анализа по методу главных компонент;
5. Проверки гипотез и статистической выборки.

Общие сведения о выбранном алгоритме машинного обучения

Random forest (с англ. — «случайный лес») — алгоритм машинного обучения, предложенный Лео Брейманом и Адель Катлер, заключающийся в использовании комитета (ансамбля) решающих деревьев. Алгоритм сочетает в себе две основные идеи: метод бэггинга Бреймана, и метод случайных подпространств, предложенный Тин Кам Хо. Алгоритм применяется для задач классификации, регрессии и кластеризации. Основная идея заключается в использовании большого ансамбля решающих деревьев, каждое из которых само по себе даёт очень невысокое качество классификации, но за счёт их большого количества результат получается хорошим.

Достоинства:

- Способность эффективно обрабатывать данные с большим числом признаков и классов.
- Нечувствительность к масштабированию значений признаков.
- Одинаково хорошо обрабатываются как непрерывные, так и дискретные признаки. Существуют методы построения деревьев по данным с пропущенными значениями признаков.
- Существуют методы оценивания значимости отдельных признаков в модели.
- Внутренняя оценка способности модели к обобщению.
- Высокая параллелизуемость и масштабируемость.

Недостатки:

- Большой размер получающихся моделей. Требуется $O\ (K)$ памяти для хранения модели, где K — число деревьев.

Ход работы

В процессе работы мы рассмотрим набор данных, состоящий из популярных детских имен. Сперва мы подключим контекст Spark, а также укажем в качестве dataframe, описанный выше набор данных. Дополнительно преобразуем имена детей в нижний регистр, что позволит нам избавиться от повторяющихся значений. Первые 20 строк набора показаны на рисунке 1.

```
+-------------+------+--------------------------+------------------+-----+----+
|Year of Birth|Gender|Ethnicity                 |Child's First Name|Count|Rank|
+-------------+------+--------------------------+------------------+-----+----+
|2016         |FEMALE|ASIAN AND PACIFIC ISLANDER|olivia            |172  |1   |
|2016         |FEMALE|ASIAN AND PACIFIC ISLANDER|chloe             |112  |2   |
|2016         |FEMALE|ASIAN AND PACIFIC ISLANDER|sophia            |104  |3   |
|2016         |FEMALE|ASIAN AND PACIFIC ISLANDER|emma              |99   |4   |
|2016         |FEMALE|ASIAN AND PACIFIC ISLANDER|emily             |99   |4   |
|2016         |FEMALE|ASIAN AND PACIFIC ISLANDER|mia               |79   |5   |
|2016         |FEMALE|ASIAN AND PACIFIC ISLANDER|charlotte         |59   |6   |
|2016         |FEMALE|ASIAN AND PACIFIC ISLANDER|sarah             |57   |7   |
|2016         |FEMALE|ASIAN AND PACIFIC ISLANDER|isabella          |56   |8   |
|2016         |FEMALE|ASIAN AND PACIFIC ISLANDER|hannah            |56   |8   |
|2016         |FEMALE|ASIAN AND PACIFIC ISLANDER|grace             |54   |9   |
|2016         |FEMALE|ASIAN AND PACIFIC ISLANDER|angela            |54   |9   |
|2016         |FEMALE|ASIAN AND PACIFIC ISLANDER|ava               |53   |10  |
|2016         |FEMALE|ASIAN AND PACIFIC ISLANDER|joanna            |49   |11  |
|2016         |FEMALE|ASIAN AND PACIFIC ISLANDER|amelia            |44   |12  |
|2016         |FEMALE|ASIAN AND PACIFIC ISLANDER|evelyn            |42   |13  |
|2016         |FEMALE|ASIAN AND PACIFIC ISLANDER|ella              |42   |13  |
|2016         |FEMALE|ASIAN AND PACIFIC ISLANDER|arya              |42   |13  |
|2016         |FEMALE|ASIAN AND PACIFIC ISLANDER|ariana            |40   |14  |
|2016         |FEMALE|ASIAN AND PACIFIC ISLANDER|maya              |39   |15  |
+-------------+------+--------------------------+------------------+-----+----+
only showing top 20 rows
```

Рисунок 1. Топ 20 строк набора данных

Поставим задачу исследовать взаимосвязь числа имен, их ранга и их классификацию по этническому происхождению. В ходе исследования мы хотим попробовать предсказать, что большая или, наоборот, меньшая часть детей принадлежит конкретно взятому этносу.

В данной работе планируется использовать алгоритм машинного обучения – случайный лес, для которого требуется сперва обозначить столбцы, которые будут использоваться в качестве функций.

В процессе работы у нас появляются трудности с объединением данных, поэтому мы используем VectorAssembler – это преобразователь, который объединяет заданный список столбцов в один векторный столбец. Это полезно для объединения необработанных функций и функций, созданных различными преобразователями функций, в один вектор функций.

Работать напрямую с данными, хоть и в случае моего небольшого набора данных, не составляет труда, но для удобства обращения воспользуемся StringIndexer, который кодирует строковый столбец меток в столбец индексов меток. Также введем столбец features, который будет агрегировать значения числа детей, их ранга и этноса. Результаты представлены на рисунке 2.

Year of Birth	Gender	Ethnicity	Child's First Name	Count	Rank	indexEthnicity	features	label
2016	FEMALE	ASIAN AND PACIFIC ISLANDER	olivia	172	1	3.0	[3.0,172.0,1.0]	3.0
2016	FEMALE	ASIAN AND PACIFIC ISLANDER	chloe	112	2	3.0	[3.0,112.0,2.0]	3.0
2016	FEMALE	ASIAN AND PACIFIC ISLANDER	sophia	104	3	3.0	[3.0,104.0,3.0]	3.0
2016	FEMALE	ASIAN AND PACIFIC ISLANDER	emma	99	4	3.0	[3.0,99.0,4.0]	3.0
2016	FEMALE	ASIAN AND PACIFIC ISLANDER	emily	99	4	3.0	[3.0,99.0,4.0]	3.0
2016	FEMALE	ASIAN AND PACIFIC ISLANDER	mia	79	5	3.0	[3.0,79.0,5.0]	3.0
2016	FEMALE	ASIAN AND PACIFIC ISLANDER	charlotte	59	6	3.0	[3.0,59.0,6.0]	3.0
2016	FEMALE	ASIAN AND PACIFIC ISLANDER	sarah	57	7	3.0	[3.0,57.0,7.0]	3.0
2016	FEMALE	ASIAN AND PACIFIC ISLANDER	isabella	56	8	3.0	[3.0,56.0,8.0]	3.0
2016	FEMALE	ASIAN AND PACIFIC ISLANDER	hannah	56	8	3.0	[3.0,56.0,8.0]	3.0
2016	FEMALE	ASIAN AND PACIFIC ISLANDER	grace	54	9	3.0	[3.0,54.0,9.0]	3.0
2016	FEMALE	ASIAN AND PACIFIC ISLANDER	angela	54	9	3.0	[3.0,54.0,9.0]	3.0
2016	FEMALE	ASIAN AND PACIFIC ISLANDER	ava	53	10	3.0	[3.0,53.0,10.0]	3.0
2016	FEMALE	ASIAN AND PACIFIC ISLANDER	joanna	49	11	3.0	[3.0,49.0,11.0]	3.0
2016	FEMALE	ASIAN AND PACIFIC ISLANDER	amelia	44	12	3.0	[3.0,44.0,12.0]	3.0
2016	FEMALE	ASIAN AND PACIFIC ISLANDER	evelyn	42	13	3.0	[3.0,42.0,13.0]	3.0
2016	FEMALE	ASIAN AND PACIFIC ISLANDER	ella	42	13	3.0	[3.0,42.0,13.0]	3.0
2016	FEMALE	ASIAN AND PACIFIC ISLANDER	arya	42	13	3.0	[3.0,42.0,13.0]	3.0
2016	FEMALE	ASIAN AND PACIFIC ISLANDER	ariana	40	14	3.0	[3.0,40.0,14.0]	3.0
2016	FEMALE	ASIAN AND PACIFIC ISLANDER	maya	39	15	3.0	[3.0,39.0,15.0]	3.0

only showing top 20 rows

Рисунок 2. Результаты агрегирования данных

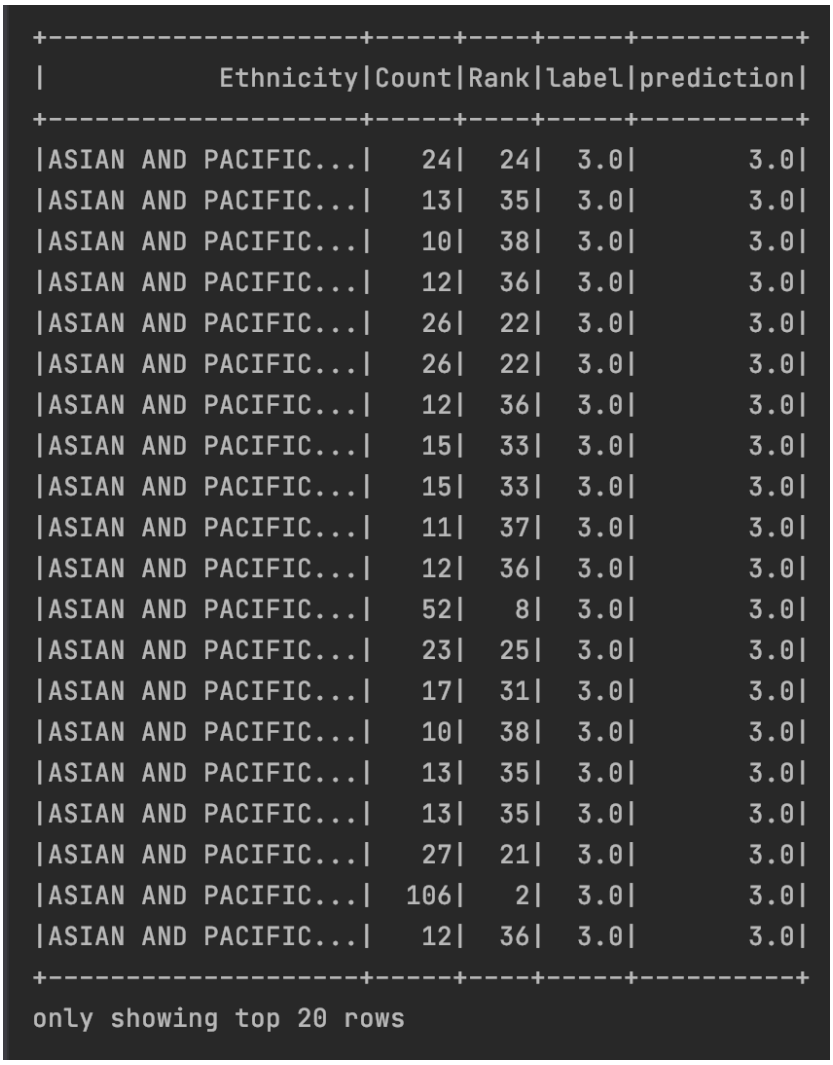
Определим данные, которые мы ищем, дополнительно переименовав значения со стандартами Spark MLlib. Ввиду наличия большого объема данных, нам представляется возможным разбить их на более мелкие части. Таким образом, мы подготовим данные для случайного леса.

Определим изначальное количество данных в наборе, а также число в обучающую выборку и в тестовую. Результаты продемонстрируем на рисунке 3.

```
dataframe count: 19418
training count: 13599
test count: 5819
```

Рисунок 3. Число данных по выборкам

Зададим необходимые параметры для обучения и поддержания точности на уровне, указанном в цели работы. Полученные результаты отобразим на рисунке 4.



Ethnicity	Count	Rank	label	prediction
ASIAN AND PACIFIC...	24	24	3.0	3.0
ASIAN AND PACIFIC...	13	35	3.0	3.0
ASIAN AND PACIFIC...	10	38	3.0	3.0
ASIAN AND PACIFIC...	12	36	3.0	3.0
ASIAN AND PACIFIC...	26	22	3.0	3.0
ASIAN AND PACIFIC...	26	22	3.0	3.0
ASIAN AND PACIFIC...	12	36	3.0	3.0
ASIAN AND PACIFIC...	15	33	3.0	3.0
ASIAN AND PACIFIC...	15	33	3.0	3.0
ASIAN AND PACIFIC...	11	37	3.0	3.0
ASIAN AND PACIFIC...	12	36	3.0	3.0
ASIAN AND PACIFIC...	52	8	3.0	3.0
ASIAN AND PACIFIC...	23	25	3.0	3.0
ASIAN AND PACIFIC...	17	31	3.0	3.0
ASIAN AND PACIFIC...	10	38	3.0	3.0
ASIAN AND PACIFIC...	13	35	3.0	3.0
ASIAN AND PACIFIC...	13	35	3.0	3.0
ASIAN AND PACIFIC...	27	21	3.0	3.0
ASIAN AND PACIFIC...	106	2	3.0	3.0
ASIAN AND PACIFIC...	12	36	3.0	3.0

only showing top 20 rows

Рисунок 4. Результаты предсказаний

Мы можем увидеть, что теперь наша модель может предсказать по рангу и числу вероятный этнос ребенка. Крайний правый столбец, если исходить из наших предложений, совпадает с действительным значением, что является верным результатом исследования. Наглядной демонстрацией данного вывода является сходство значений столбцов label и prediction.

Отдельно посчитаем точность наших предсказаний и округлим значение до двух знаков после запятой, что будет представлено на рисунке 5.

Рисунок 5.

Листинг кода

```
package LabFour

import org.apache.log4j.Level.WARN
import org.apache.log4j.LogManager
import org.apache.spark.ml.classification.{RandomForestClassificationModel, RandomForestClassifier}
import org.apache.spark.ml.evaluation.MulticlassClassificationEvaluator
import org.apache.spark.ml.feature.{StringIndexer, VectorAssembler}
import org.apache.spark.sql.functions.{col, lower}
import org.apache.spark.sql.{DataFrame, SparkSession}

object LabFour {
  val PATH: String = "src/main/data"
  val NODES: Int = 3

  def main(args: Array[String]): Unit = {
    val spark: SparkSession = SparkSession
      .builder()
      .appName("Lab4")
      .master(s"local[$NODES]")
      .getOrCreate
    LogManager.getRootLogger.setLevel(WARN)

    val dataframe: DataFrame = spark
      .read
      .format("csv")
      .option("header", "true")
      .option("delimiter", ",")
      .option("inferSchema", value = true)
      .load(s"$PATH/var.csv")
      .withColumn("Child's First Name", lower(col("Child's First Name")))

    dataframe.show(false)

    // Indexing some columns for using as features
    val ethnicity: DataFrame = new StringIndexer()
      .setInputCol("Ethnicity")
      .setOutputCol("indexEthnicity")
      .fit(dataframe)
      .transform(dataframe)

    // Vectorization of required columns
    val cols: Array[String] = Array("indexEthnicity", "Count", "Rank")
    val assembler: VectorAssembler = new VectorAssembler()
      .setInputCols(cols)
      .setOutputCol("features")
    val feature: DataFrame = assembler
      .transform(ethnicity)

    // Renaming columns for suiting SparkML
```

```
    val indexer: StringIndexer = new StringIndexer()
      .setInputCol("Ethnicity")
      .setOutputCol("label")
    val label: DataFrame = indexer
      .fit(feature)
      .transform(feature)
    label.show(false)

    // Splitting dataframe into training and test dataframes
    val seed: Int = 5043
    val Array(training, test) = label.randomSplit(Array(0.7, 0.3), seed)

    println(s"dataframe count: ${dataframe.count()}")
    println(s"training count: ${training.count()}")
    println(s"test count: ${test.count()}\n")

    // Train RandomForestClassifier model with training data set
    // Setting max feature bins at 330
    val regression: RandomForestClassifier = new RandomForestClassifier()
      .setLabelCol("label")
      .setFeaturesCol("features")
      .setMaxBins(330)
    val model: RandomForestClassificationModel = regression
      .fit(training)

    // Run model with test data set to get predictions
    // This will add new columns rawPrediction, probability and prediction
    val prediction: DataFrame = model
      .transform(test)
    prediction
      .select("Ethnicity", "Count", "Rank", "label", "prediction")
      .show(20)

    // Select (prediction, label) and compute accuracy.
    val evaluator: MulticlassClassificationEvaluator = new MulticlassClassificationEvaluator()
      .setLabelCol("label")
      .setPredictionCol("prediction")
      .setMetricName("accuracy")
    val accuracy: Double = evaluator
      .evaluate(prediction)

    // Round accuracy up to 2 digits
    println(
      s"Accuracy = ${
        BigDecimal(accuracy)
          .setScale(2, BigDecimal.RoundingMode.HALF_UP)
      }"
    )
  }
}
```

Вывод

В ходе данной лабораторной работы я изучил особенности и возможности способов работы со Spark ML и предсказания данных на основе случайного леса. Для этого я использовал набор данных популярных детских имен. Полученная модель умеет по числу детей, которые родились в определенный год и по рангу имени, определять этнос ребенка. Данное исследование позволяет выявить наибольшее или наименьшее число детей, которые относятся к выбранному этносу. Сами значения могут быть использованы, например, при изготовке вакцин. Так, при росте числа новорожденных в Нью-Йорке следует изготовлять вакцину с большей вероятностью для определённой этнической группы.