



Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования

«Московский государственный технический университет имени
Н.Э. Баумана (национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ «Информатика и системы управления»
КАФЕДРА «Компьютерные системы и сети (ИУ-6)»

ОТЧЕТ О ВЫПОЛНЕНИИ ЛАБОРАТОРНОЙ РАБОТЫ

по дисциплине «Большие данные: инструменты и
технологии»

Студент:	Козлов Владимир Михайлович
Группа:	ИУ6-33М
Тип задания:	лабораторная работа
Тема:	Решение задач классификации и прогноза с помощью Apache ML

Студент

подпись, дата

Козлов В.М.

Фамилия, И.О.

Преподаватель

подпись, дата

Григоренко В.М.

Фамилия, И.О.

Москва, 2025

Содержание

Задание	3
1 Ход работы	3
1.1 Классификация	3
1.2 Регрессия	5
2 Вывод	8

Задание

Цель

1. Освоить основные этапы применения алгоритмов машинного обучения в Apache Spark MLlib или Flink ML для решения задач классификации и регрессии (прогнозирования).
2. Научиться работать с классическими открытыми датасетами.
3. Закрепить изученный материал (методы обработки, обучения, тестирования, оценки метрик).

1 Ход работы

Для выполнения использовался тот же docker-образ, что и для ЛР 1 и 2.

Перед запуском на Hadoop загружаются файлы датасетов.

Листинг 1: скрипт загрузки датасетов

```
1 hdfs dfs -mkdir /data
2 hdfs dfs -put iris.csv /data/iris.csv
3 hdfs dfs -put winequality-red.csv /data/winequality-red.csv
4 hdfs dfs -put winequality-white.csv /data/winequality-white.csv
```

1.1 Классификация

От преподавателя был получен вариант 1 по классификации - Классификация видов ирисов (3 класса, 4 признака). В файл по предоставленной ссылке был добавлен хедер "X1,X2,X3,X4,Y" для работы с ним через pyspark.

Код скрипта классификации представлен ниже.

Листинг 2: Скрипт классификации

```
1 #!/bin/python3
2 from pyspark.sql import SparkSession
3 from pyspark.ml.feature import VectorAssembler, StandardScaler
4 from pyspark.ml.classification import LogisticRegression
5 from pyspark.ml import Pipeline
6 from pyspark.ml.evaluation import
    MulticlassClassificationEvaluator
7 from pyspark.ml.feature import StringIndexer
8
9 try:
10     client =
        SparkSession.builder.appName("classifier").getOrCreate()
```

```

11     df_iris = client.read.csv(
12         path="hdfs://localhost:9000/data/iris.csv",
13         header=True,
14         inferSchema=True,
15     )
16
17     df_iris.printSchema()
18     df_iris.show(3)
19
20     assembler = VectorAssembler(
21         inputCols=["X1", "X2", "X3", "X4"], outputCol="features"
22     )
23     indexer = StringIndexer(inputCol="Y", outputCol="label")
24     scaler = StandardScaler(inputCol="features",
25                             outputCol="scaledFeatures")
26
27     train_data, test_data = df_iris.randomSplit([0.8, 0.2],
28         seed=42)
29     lr = LogisticRegression(featuresCol="scaledFeatures",
30                             labelCol="label")
31
32     pipeline_lr = Pipeline(stages=[assembler, indexer, scaler,
33                                   lr])
34     model_lr = pipeline_lr.fit(train_data)
35
36     predictions_lr = model_lr.transform(test_data)
37     evaluator =
38     MulticlassClassificationEvaluator(labelCol="label",
39                                     predictionCol="prediction")
40     accuracy_lr = evaluator.evaluate(predictions_lr,
41                                     {evaluator.metricName: "accuracy"})
42
43     print(f"Logistic Regression Accuracy: {accuracy_lr:.2f}")
44 finally:
45     if client:
46         client.stop()

```

После загрузки датасета выводятся его первые 3 строки и схема для проверки соответствия датасета желаемому. Далее все признаки объединяются в один вектор, а строковые метки преобразуются в числовые. Затем через StandartScaler проводится масштабирование. После чего датасет делится на обучающую и тестовую выборку, инициализируется модель логистической регрессии. На последнем этапе проверяется точность предсказания.

Вывод скрипта представлен ниже.

Листинг 3: Вывод скрипта классификации

```

root
|-- X1: double (nullable = true)
|-- X2: double (nullable = true)
|-- X3: double (nullable = true)
|-- X4: double (nullable = true)
|-- Y: string (nullable = true)

```

```

+---+---+---+---+-----+
| X1| X2| X3| X4|          Y|
+---+---+---+---+-----+
|5.1|3.5|1.4|0.2|Iris-setosa|
|4.9|3.0|1.4|0.2|Iris-setosa|
|4.7|3.2|1.3|0.2|Iris-setosa|
+---+---+---+---+-----+
only showing top 3 rows

```

Logistic Regression Accuracy: 1.00

Как видно в выводе точность получилась 1, то есть модель обучилась верно.

1.2 Регрессия

От преподавателя был получен вариант 1 по регрессии - Физико-химические параметры и качество португальских вин.

Код скрипта регрессии представлен ниже.

Листинг 4: Скрипт регрессии

```

1 #!/bin/python3
2 from pyspark.sql import SparkSession
3 from pyspark.ml.feature import VectorAssembler, StandardScaler,
  PCA
4 from pyspark.ml.regression import LinearRegression,
  RandomForestRegressor
5 from pyspark.ml import Pipeline
6 from pyspark.ml.evaluation import RegressionEvaluator
7
8 client = None
9
10 try:
11     client = SparkSession.builder.appName("regress").getOrCreate()
12     df_wine_red = client.read.csv(
13         path="hdfs://localhost:9000/data/winequality-red.csv",
14         header=True,
15         inferSchema=True,
16         sep=";",
17     )

```

```

18 df_wine_white = client.read.csv(
19     path="hdfs://localhost:9000/data/winequality-white.csv",
20     header=True,
21     inferSchema=True,
22     sep=";",
23 )
24 df = df_wine_red.union(df_wine_white)
25
26 df.printSchema()
27 df.show(3)
28
29 assembler = VectorAssembler(
30     inputCols=["fixed acidity", "volatile acidity", "citric
acid", "residual sugar", "chlorides", "free sulfur dioxide",
"total sulfur dioxide", "density", "pH", "sulphates",
"alcohol"],
31     outputCol="features",
32 )
33 pca = PCA(k=6, inputCol="features", outputCol="pcaFeatures")
34 train_data, test_data = df.randomSplit([0.8, 0.2], seed=42)
35 rf = RandomForestRegressor(
36     featuresCol="pcaFeatures",
37     labelCol="quality",
38     numTrees=100,
39     maxDepth=10,
40     seed=42
41 )
42
43 pipeline_reg = Pipeline(stages=[assembler, pca, rf])
44 model_reg = pipeline_reg.fit(train_data)
45 predictions_reg = model_reg.transform(test_data)
46
47 evaluator_reg = RegressionEvaluator(labelCol="quality",
predictionCol="prediction")
48 rmse = evaluator_reg.evaluate(predictions_reg,
{evaluator_reg.metricName: "rmse"})
49 mae = evaluator_reg.evaluate(predictions_reg,
{evaluator_reg.metricName: "mae"})
50
51 print(f"Linear Regression RMSE: {rmse:.2f}, MAE: {mae:.2f}")
52 finally:
53     if client:
54         client.stop()

```

Для уменьшения размерности признаков был выбран PCA, который оставляет 6 наиболее информативных признаков. В качестве модели выбран случайный лес, в ко-

тором 100 деревьев глубины 10. Для оценки качества работы модели были выбраны RMSE и MAE.

Вывод скрипта регрессии представлен ниже.

Листинг 5: Вывод скрипта регрессии

```
smaller than tiny
root
-- fixed acidity: double (nullable = true)
-- volatile acidity: double (nullable = true)
-- citric acid: double (nullable = true)
-- residual sugar: double (nullable = true)
-- chlorides: double (nullable = true)
-- free sulfur dioxide: double (nullable = true)
-- total sulfur dioxide: double (nullable = true)
-- density: double (nullable = true)
-- pH: double (nullable = true)
-- sulphates: double (nullable = true)
-- alcohol: double (nullable = true)
-- quality: integer (nullable = true)

+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
|fixed acidity|volatile acidity|citric acid|residual sugar|chlorides|free sulfur dioxide|total sulfur dioxide|density|-----+-----+-----+
|↪ pH|sulphates|alcohol|quality|-----+-----+-----+-----+-----+-----+-----+-----+
|↪ 7.4|5|0.7|0.0|1.9|0.076|11.0|34.0|0.9978|3.51|0.56|9.4|
|↪ 7.8|5|0.88|0.0|2.6|0.098|25.0|67.0|0.9968|3.2|0.68|9.8|
|↪ 7.8|5|0.76|0.04|2.3|0.092|15.0|54.0|0.997|3.26|0.65|9.8|
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
only showing top 3 rows
Linear Regression RMSE: 0.69, MAE: 0.54
```

Показания довольно высоки, то есть модель часто ошибается. Получается, модель не очень подходит для этой задачи.

2 Вывод

в ходе выполнения лабораторной работы был закреплён полученный материал: освоены основные этапы применения алгоритмов машинного обучения в Apache Spark MLlib для решения задач классификации и регрессии, получены навыки работы с классическими открытыми датасетами.