Sveučilište u Zagrebu
Fakultet elektrotehnike i računarstva
Zavod za elektroniku, mikroelektroniku, računalne i inteligentne sustave

POSREDNICI UMREŽENIH SUSTAVA

Ak. god. 2021./2022.

2. laboratorijska vježba: Apache Spark

autor: Davor Vukadin (davor.vukadin@fer.hr)

Ova laboratorijska vježba provest će vas kroz osnove korištenja i radni primjer učenja i validacije modela strojnog učenja kroz radni okvir Apache Spark.

Iako to nije nužno, preporučujemo upotrebu Linux operacijskog sustava te su i ove upute prilagođene njemu.

1. Instalacija potrebnih alata

Prije svega, potrebno je na vaše računalo instalirati sljedeće:

python https://www.python.org/downloads/
pyspark https://spark.apache.org/docs/latest/api/python/getting_started/install.html
java `sudo apt install openjdk-8-jre-headless`
classic jupyter notebook https://jupyter.org/install

Uspješnu instalaciju svake od komponenti možete provjeriti izvođenjem sljedećih naredbi u terminalu:

python `python3 --version`
pyspark `pip show pyspark`
java `java -version`
classic jupyter notebook `jupyter --version`

2. Pokretanje bilježnice i izvođenje jednostavnog primjera

Pokrenite *jupyter* bilježnicu kroz terminal: `jupyter notebook`

Otvorit će se web preglednik kroz koji ćete pritiskom na tipku “*New*” stvoriti novu bilježnicu u trenutnom direktoriju.

Uvezite **pyspark** izvođenjem sljedeće naredbe u prvu ćeliju:

```
import pyspark
```

Inicijalizirajte novu sesiju unutar *pyspark* frameworka:

```
from pyspark.sql import SparkSession

spark = SparkSession.builder.appName("PUS Lab 2.").getOrCreate()
```

Izvedite jednostavan primjer *pipeline*-a i učenja modela logističke regresije na klasifikaciji teksta:

1) Uvezivanje potrebnih komponenti:

```
from pyspark.ml import Pipeline
from pyspark.ml.classification import LogisticRegression
from pyspark.ml.feature import HashingTF, Tokenizer
```

2) Kreacija *DataFrame* objekata za učenje i testiranje modela - u ovom slučaju zadatak modela je predvidjeti nalazi li se u nizu riječ “spark”:

```
# Prepare training documents from a list of (id, text, label) tuples.
training = spark.createDataFrame([
    (0, "a b c d e spark", 1.0),
    (1, "b d", 0.0),
    (2, "spark f g h", 1.0),
    (3, "hadoop mapreduce", 0.0)
], ["id", "text", "label"])


# Prepare test documents, which are unlabeled (id, text) tuples.
test = spark.createDataFrame([
    (4, "spark i j k"),
    (5, "l m n"),
    (6, "spark hadoop spark"),
    (7, "apache hadoop")
], ["id", "text"])
```

3) Definicija *pipeline*-a:
   a) Tokenizer - razdvaja rečenice na riječi
   b) HashingTF - pretvara setove riječi iz rečenice u vektore fiksne veličine (brojčana reprezentacija rečenice)
   c) LogisticRegression - model koji učimo - logistička regresija
   d) Pipeline - pipeline objekt kojim se definira slijed izvođenja učenja modela

```
tokenizer = Tokenizer(inputCol="text", outputCol="words")
hashingTF = HashingTF(inputCol=tokenizer.getOutputCol(),
outputCol="features")
lr = LogisticRegression(maxIter=10, regParam=0.001)
pipeline = Pipeline(stages=[tokenizer, hashingTF, lr])
```

4) Učenje modela na podatcima za učenje:

```
model = pipeline.fit(training)
```

5) Izvršavanje predikcija na testnom skupu i njihov ispis:

```
prediction = model.transform(test)
selected = prediction.select("id", "text", "probability", "prediction")
for row in selected.collect():
    rid, text, prob, prediction = row
    print(
        "(%d, %s) --> prob=%s, prediction=%f" % (
            rid, text, str(prob), prediction
        )
    )
```

3. Zadatak: učenje modela za klasifikaciju sentimenta osvrta na film

Vaš je zadatak za ovu laboratorijsku vježbu naučiti model klasifikacije sentimenta s obzirom na dani osvrt na film. Kao skup podataka koristit ćete Large Movie Review Dataset (https://ai.stanford.edu/~amaas/data/sentiment/), koji sadrži 25 000 primjera za učenje i 25 000 primjera za testiranje modela. Vježba je podijeljena u nekoliko dijelova:

1. *(2.5 bodova)* Preuzimanje skupa podataka, njegovo predprocesiranje (uklanjanje svih nealfanumeričkih znakova te pretvaranje cijelog teksta u mala slova) te izrada **DataFrame** objekata za učenje i testiranje modela. Paralelizirajte **RDD** koji predajete prilikom izrade DataFrame objekta na 100 *slice*-ova (**SparkContext.parallelize**).

2. *(2.5 bodova)* Izrada *pipeline*-a, slično kao u ranijem primjeru, dodatno uz uklanjanje **engleskih** stop riječi (https://en.wikipedia.org/wiki/Stop_word) pomoću klase **StopWordsRemover** unutar *pipeline*-a, te vektorizaciju pomoću klase **CountVectorizer** umjesto **HashingTF**. Pripazite na veličinu vokabulara koju dopuštate toj klasi kako zauzeće memorije ne bi bilo preveliko.

3. *(5 bodova)* Naučite i evaluirajte model logističke regresije (uz *default* parametre) pomoću metrika točnosti, preciznosti i odziva (klasa **MulticlassMetrics**) na podatcima za učenje, odnosno testiranje. Koliko je *false-negative*, odnosno *false-positive* ishoda napravio model prilikom evaluacije na testnom skupu?

4. *(2.5 bodova)* Umjesto korištenja samo jedne riječi prilikom izrade vektora značajki koje ulaze u model, iskoristite 2-grame (https://en.wikipedia.org/wiki/N-gram) (klasa **NGram**) ulaznih rečenica kao ulaz u vektorizaciju te ponovno evaluirajte model. Jesu li rezultati bolji nego prije?

5. *(2.5 bodova)* Spojite vektore značajki jednorječne vektorizacije i 2-gram vektorizacije koje ulaze u model pomoću **VectorAssembler**. Ponovno evaluirajte model i usporedite rezultate.

6. *(2.5 bodova)* Umjesto logističke regresije naučite modele SVM-a (**LinearSVC**) te model naivnog Bayesa (**NaiveBayes**) te usporedite rezultate evaluacije s logističkom regresijom.