2018

Semestrální práce

BI-BIG

ADAM SKLUZÁČEK

OBSAH

DATASETY

Původní datasety byly stažené z internetové filmové databáze IMDb, konkrétně z adresy imdb.com/interfaces byly staženy tyto datasety:

1. *title.basics.tsv*
2. *title.crew.tsv*
3. *title.ratings.tsv*
4. *name.basics.tsv*

Datasety obsahují miliony záznamů, pro účely této úlohy byly tedy vhodným způsobem transformovány. Z datasetů byly odstraněny řádky obsahující null hodnoty, upravené některé sloupce a datasetu *title.basics.tsv* byl přidán sloupec *director_id* z *title.crew.tsv*. Kromě těchto transformací bylo potřeba provést join datasetů, oříznout vrchních 50000 záznamů a znovu datasety rozdělit. Tento krok byl nutný, vzhledem k tomu, že původní datasety obsahovaly miliony záznamů. Pokud bychom rovnou ořízli vrchních 50000 řádků z původních datasetů a tyto data agregovali, pak by nám pro většinu titulů chyběli údaje o režisérech. Veškeré transformace si můžete prohlédnout v přiloženém jupyter notebooku *transformations.ipytnb*. Výsledkem transformací jsou tyto 3 datasety:

title_basics.csv

Obsahuje základní informace o jednotlivých titulech, konkrétně tyto sloupce:

1. **title_id** (string) - jednoznačný identifikátor titulu
2. **titleType** (string) - určuje typ titulu, může nabývat hodnot např. *short* (= krátkometrážní film), *movie* (=film), *tvSeries* (= seriál) atd.
3. **primaryTitle** (string) – primární název titulu, tedy nejpopulárnější název daného titulu, nebo název, který byl použit nejčastěji na propagačních materiálech
4. **originalTitle** (string) – originální název titulu, tedy název titulu v původním jazyce
5. **isAdult** (boolean) – určuje, zda se jedná o titul určený pouze pro dospělé, může nabývat hodnot 0 (= titul není určený pouze pro dospělé) nebo 1 (= titul je určený pouze pro dospělé)
6. **releaseYear** (integer) – rok, ve kterém byl daný titul uveden
7. **runtimeMinutes** (integer) – délka titulu v minutách
8. **genre** (string) – hlavní žánr titulu, může nabývat hodnot např. *document*, *comedy*, *romance* atd.
9. **director_id** (string) – obsahuje jednoznačný identifikátor osoby, která daný titul režírovala

Ukázka prvních 10 záznamů včetně hlavičky:

	title_id	titleType	primaryTitle	originalTitle	isAdult	releaseYear	runtimeMinutes	genre	director_id
0	tt0000001	short	Carmencita	Carmencita	0	1894	1	Documentary	nm0005690
1	tt0000002	short	Le clown et ses chiens	Le clown et ses chiens	0	1892	5	Animation	nm0721526
2	tt0000003	short	Pauvre Pierrot	Pauvre Pierrot	0	1892	4	Animation	nm0721526
3	tt0000005	short	Blacksmith Scene	Blacksmith Scene	0	1893	1	Comedy	nm0005690
4	tt0000006	short	Chinese Opium Den	Chinese Opium Den	0	1894	1	Short	nm0005690
5	tt0000008	short	Edison Kinetoscopic Record of a Sneeze	Edison Kinetoscopic Record of a Sneeze	0	1894	1	Documentary	nm0005690
6	tt0000009	movie	Miss Jerry	Miss Jerry	0	1894	45	Romance	nm0085156
7	tt0000010	short	Exiting the Factory	La sortie de l'usine Lumière à Lyon	0	1895	1	Documentary	nm0525910
8	tt0000011	short	Akrobatisches Potpourri	Akrobatisches Potpourri	0	1895	1	Documentary	nm0804434
9	tt0000013	short	The Photographical Congress Arrives in Lyon	Neuville-sur-Saône: Débarquement du congrès de...	0	1895	1	Documentary	nm0525910

title_ratings.csv

Obsahuje hodnocení uživatelů pro jednotlivé tituly, konkrétně tyto sloupce:

1. **title_id** (string) - jednoznačný identifikátor titulu
2. **averageRating** (float) – průměrné hodnocení uživatelů daného titulu, může nabývat hodnot od 1 do 10
3. **numVotes** (integer) – počet hodnocení uživatelů daného titulu

Ukázka prvních 10 záznamů včetně hlavičky:

	title_id	averageRating	numVotes
0	tt0000001	5.8	1444
1	tt0000002	6.4	174
2	tt0000003	6.6	1045
3	tt0000005	6.2	1742
4	tt0000006	5.5	93
5	tt0000008	5.6	1542
6	tt0000009	5.6	74
7	tt0000010	6.9	5147
8	tt0000011	5.4	215
9	tt0000013	5.7	1326

director_basics.csv

Obsahuje základní informace o režisérech (z datasetu name_basics.csv byli vybráni pouze režiséři některého z filmů v title_basics.csv). Konkrétně obsahuje tyto sloupečky:

1. **director_id** (string) – jednoznačný identifikátor osoby (v našem případě jsou všechny osoby režiséry)
2. **director_primaryName** (string) – jméno, pod kterým je nejčastěji režisér uváděn
3. **director_birthYear** (integer) – rok narození režiséra
4. **director_deathYear** (integer) – rok úmrtí režiséra, v našem případě neobsahuje žádné null hodnoty, null hodnota by znamenala, že se jedná o stále žijícího režiséra
5. **director_primaryProfession** (string) – profese, kterou se daný režisér nejvíce proslavil, může nabývat hodnot např. director (= režisér), producer (= producent) atd.
6. **director_knownForTitle** (string) – obsahuje ID titulu, kterým se daný režisér nejvíce proslavil

Ukázka prvních 10 záznamů:

	director_id	director_primaryName	director_birthYear	director_deathYear	director_primaryProfession	director_knownForTitle
0	nm0005690	William K.L. Dickson	1860	1935	cinematographer	tt1496763
1	nm0721526	Émile Reynaud	1844	1918	director	tt2184231
2	nm0085156	Alexander Black	1859	1940	director	tt0000009
3	nm0525910	Louis Lumière	1864	1948	producer	tt1736627
4	nm0804434	Max Skladanowsky	1863	1939	director	tt7874452
5	nm0010291	Birt Acres	1854	1918	cinematographer	tt0000025
6	nm0617588	Georges Méliès	1861	1938	director	tt0002113
7	nm0895515	Gabriel Veyre	1871	1936	director	tt0425370
8	nm0234288	Aurélio da Paz dos Reis	1862	1931	director	tt0138659
9	nm0349785	Alice Guy	1873	1968	director	tt0003365

Datasety jsou dostupné ke stažení na následujících adresách:

- https://www.dropbox.com/s/u3eha28h0siaxlo/title_basics.csv
- https://www.dropbox.com/s/ufib27ehl3qz8ia/title_ratings.csv
- https://www.dropbox.com/s/i4vqm7kz68m5u79/director_basics.csv

IMPORT DAT A AGREGACE

HDFS a Spark byly spuštěny podle návodu v 5. cvičení. Konfigurační soubory lze nalézt v příloze v podsložce HDFS_Spark a postup byl následující (všechny příkazy byly prováděny pod rootem):

Nejprve byl vytvořen docker image Sparku:

```
docker build -f spark.df -t spark .
```

Který byl následně použit pro spuštění dvou kontejnerů (na pozadí), respektive jednoho Spark Mastera a jednoho Spark Workera příkazem:

```
docker-compose up -d
```

Pro přístup k Spark Masterovi dále potřebujeme nějaký Driver, v našem případě se bude jednat o spark-shell, který spustíme příkazem:

```
docker run -it -p 8088:8088 -p 8042:8042 -p 4041:4040 --name driver -h driver
spark:latest bash
```

Dále potřebujeme spark-shell připojit na našeho Spark Mastera, k tomu potřebujeme jistit jeho IP adresu pomocí příkazu (mimo terminálové okno, ve ktrém běží výše spuštěný shell):

```
docker inspect --format '{{ .NetworkSettings.IPAddress }}' hdfs_spark_spark-master_1
```

Následně se ze shellu připojíme na Spark Mastera tímto příkazem (použijeme IP adresu z předchozího příkazu):

```
spark-shell --master spark://172.17.0.2:7077
```

Tímto máme spuštěný Spark a zbývá nám spustit HDFS. Spustíme předkonfigurovaný Apache Hadoop se službou HDFS následujícím příkazem:

```
docker run --name hadoop -t -i sequenceiq/hadoop-docker /etc/bootstrap.sh -bash
```

Dále pomocí příkazu `ifconfig` zjistíme IP adresu HDFS. HDFS umožňuje přístup k souborům přes url, naše data uložená na HDFS budou tedy přístupná na `hdfs://172.17.0.5:9000/`

IMPORT DAT

Data do HDFS byla naimportována následujícími příkazy:

```
curl -L https://www.dropbox.com/s/u3eha28h0siaxlo/title_basics.csv --output
title_basics.csv

curl -L https://www.dropbox.com/s/ufib27ehl3qz8ia/title_ratings.csv --output
title_ratings.csv

curl -L https://www.dropbox.com/s/i4vqm7kz68m5u79/director_basics.csv --output
director_basics.csv
```

A datasety přesunuty do složky /datasets/ pomocí příkazu `hdfs dfs -put`

Následně byly datasety přesunuty do složky datasets a naimportovány z HDFS do Sparku pomocí příkazů:

```
val title_basics = spark.sqlContext.read.format("csv").option("header",
"true").option("inferSchema",
"true").load("hdfs://172.17.0.5:9000/datasets/title_basics.csv")

val title_ratings = spark.sqlContext.read.format("csv").option("header",
"true").option("inferSchema",
"true").load("hdfs://172.17.0.5:9000/datasets/title_ratings.csv")

val director_basics = spark.sqlContext.read.format("csv").option("header",
"true").option("inferSchema",
"true").load("hdfs://172.17.0.5:9000/datasets/director_basics.csv")
```

1. AGREGACE

Vytvoříme nový dataset, obsahující počet titulů pro každý typ titulu následujícím způsobem:

```
val count_titleType = title_basics.groupBy("titleType").count()
```

2. AGREGACE

Vytvoříme nový dataset, obsahující počet režírovaných titulů pro každého režiséra podle jeho jména:

```
val count_directed_titles = title_basics.join(director_basics,
"director_id").groupBy("director_primaryName").count()
```

3. AGREGACE

K předchozímu datasetu přidáme sloupec s průměrným hodnocením titulů pro každého režiséra a následně získáme nejvyšší průměrné hodnocení titulů podle počtů titulů natočených stejným režisérem (transformace je rozdělena na dvě části pro o trochu lepší čitelnost):

```
val count_avg_directed_titles = title_basics.join(director_basics,
"director_id").join(title_ratings,
"title_id").groupBy("director_primaryName").avg("averageRating").join(count_directed
_titles, "director_primaryName")


val maxavg_count_avg_directed_titles =
count_avg_directed_titles.groupBy("count").max("avg(averageRating)")
```

Následně tento dataset uložíme na HDFS do složky /datasets/final_agg_result/:

```
maxavg_count_avg_directed_titles.coalesce(1).write.option("header",
"true").csv("hdfs://172.17.0.5:9000/datasets/final_agg_result")
```

DATASET PRO ELASTICSEARCH

Jelikož v zadání není žádné omezení na dataset, který budeme nahrávat do ElasticSearche (vytvořit nad kterýmkoliv datasetem index v ElasticSearch), vytvoříme si dataset, který obsahuje data ze všech tří datasetů:

```
val dataframe = title_basics.join(title_ratings, "title_id").join(director_basics,
"director_id")
```

A výsledný dataset uložíme na HDFS do složky /datasets/:

```
dataframe.coalesce(1).write.option("header",
"true").csv("hdfs://172.17.0.5:9000/datasets/dataframe")
```

Nakonec ještě přejmenujeme uložený soubor na dataset.csv pomocí příkazu `hdfs dfs -mv`

ELASTICSEARCH

Bohužel, se mi nepodařilo propojit HDFS s Logstash pro input dat přímo z HDFS do ElasticSearch. Data jsou tedy naimportována natvrdo, ze složky datasets.

Konfigurační soubory potřebné pro spuštění potřebných kontejnerů (Logstash, ElasticSearch, Kibana) byly staženy z 9. cvičení, konkrétně z URL: https://drive.google.com/file/d/1PqEtoRUxRjWXWkQQOR-20ltMh6DY7MOi/view

Konfigurační soubory lze nalézt v příloze v podsložce ElasticSearch.

Jediná úprava (kromě přesunutí importovaného datasetu do složky logstash/datasets/) byla provedena na logstash pipelině (soubor logstash/pipeline/logstash.conf).

Byla upravena cesta k souboru s našemi daty, dále byly specifikovány jednotlivé sloupce, jako sloupec *@timestamp* byl použit sloupec *releaseYear* ve formátu yyyy a nakonec byly přetypovány sloupce obsahující číselné hodnoty.

Výsledná podoba souboru logstash/pipeline/logstash.conf:

```
input {
	file {
		path => "/datasets/dataset.csv"
		start_position => "beginning"
	}
}


filter {
		csv{
		separator => ","
		columns => ["title_id", "titleType", "primaryTitle", "originalTitle",
"isAdult", "releaseYear", "runtimeMinutes", "genre", "director_id",
"averageRating", "numVotes", "director_primaryName", "director_birthYear",
"director_deathYear", "director_primaryProfession", "director_knownForTitle"]
	}
	date{
		match => ["releaseYear", "yyyy"]
	}
	mutate {convert => ["isAdult", "boolean"]}
	mutate {convert => ["runtimeMinutes", "integer"]}
	mutate {convert => ["averageRating", "float"]}
	mutate {convert => ["numVotes", "integer"]}
	mutate {convert => ["director_birthYear", "integer"]}
	mutate {convert => ["director_deathYear", "integer"]}
}


output {
	elasticsearch {
		hosts => "http://elasticsearch:9200"
		index => "titles"
	}
}
```

Kontejnery (Logstash, ElasticSearch, Kibana) byly spuštěny na pozadí příkazem:

```
docker-compose up -d
```

Následně jsem se v prohlížeči přesunul na adresu, na které nám běží kibana: http://localhost:5601
A vytvořil Index Pattern nad titles s použitím sloupce @timestamp (rok vydání titulu) jako timestamp. Jako období jsem nastavil poslední 130 let (nejstarší titul je z roku 1892).

1. DOTAZ - FILTER

Dotaz pro vypsání všech titulů pouze pro dospělé vydaných před rokem 1980:

```
GET titles/_search
{
  "query": {
    "bool": {
      "filter": [
        { "term":  { "isAdult": true }},
        { "range": { "releaseYear": { "gte": "1980-01-01" }}}
      ]
    }
  }
}
```

2. DOTAZ - TŘÍDĚNÍ

Dotaz pro prvních 10 filmů setříděných podle hodnocení:

```
GET titles/_search
{
  "query": {
    "match": {
      "titleType": "movie"
    }
  },
  "size": 10,
  "from": 0,
  "sort":{
    "averageRating": { "order":"desc" }
  }}
```

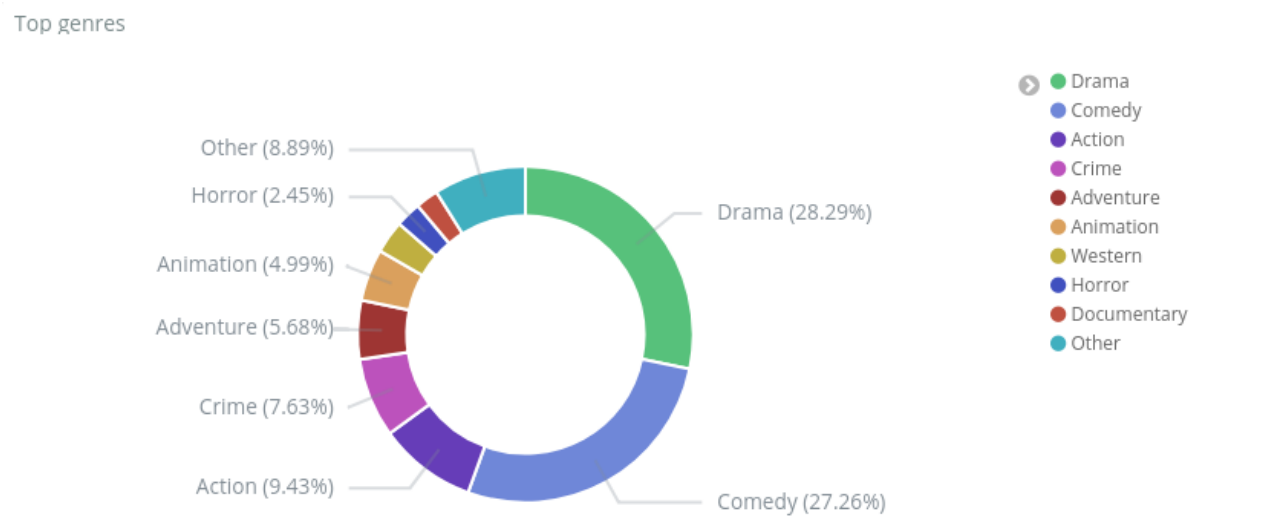
3. DOTAZ – WILDCARD

Dotaz pro všechny unikátní režiséry, jejichž křestní jméno je Louis:

```
GET titles/_search
{
  "query": {
    "wildcard": {
      "director_primaryName.keyword": "Louis*"
    }
  },
  "size": 0,
  "aggs" : {
    "names" : {
      "terms" : { "field" : "director_primaryName.keyword"}
    }
  }
}
```

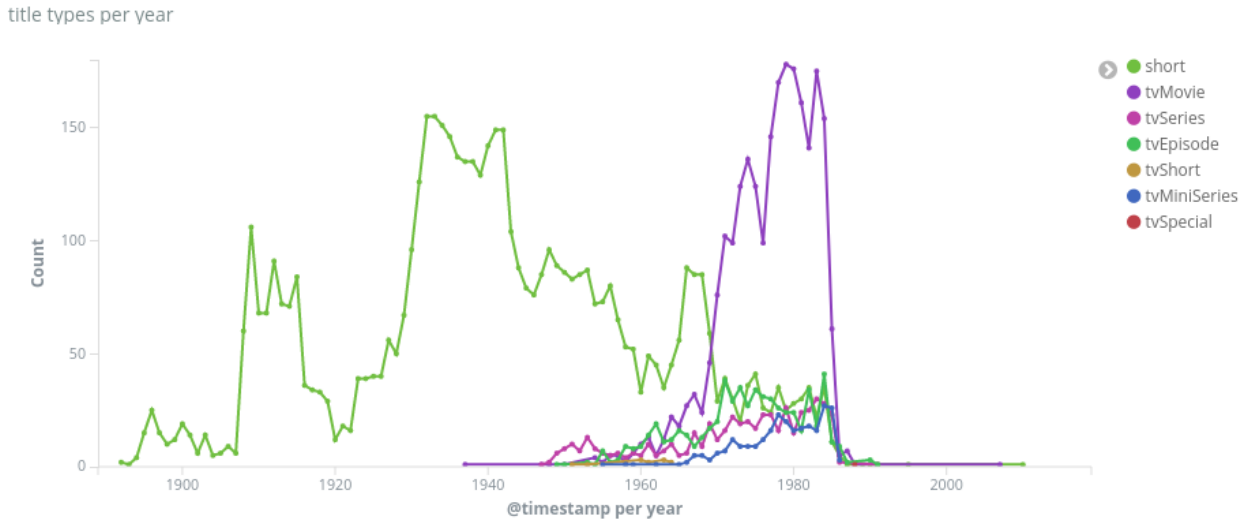
1. VIZUALIZACE – NEJČASTĚJŠÍ ŽÁNRY

Koláčový graf zobrazující 9 nejčastějších žánrů a všechny zbylé žánry spojené do jednoho:



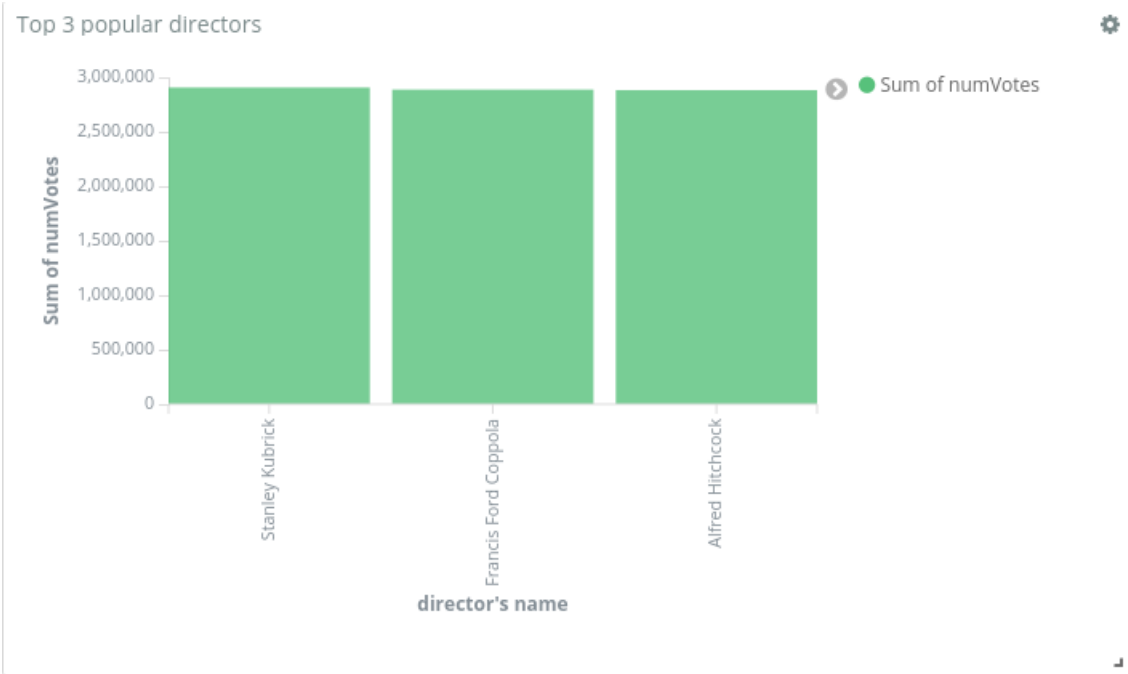
2. VIZUALIZACE – POČET TITULŮ PODLE TYPU ZA ROK

Spojnicový graf zobrazující 7 nejčastějších typů titulů za každý rok (kromě typu *movie*, který byl odfiltrován, protože byl každý rok zdaleka nejčastější a deformoval tak tvar grafu):



3. VIZUALIZACE – NEJPOPULÁRNĚJŠÍ REŽISÉŘI

Sloupcový graf zobrazující 3 nejpopulárnější režiséry (na základě součtu počtů hodnocení jejich titulů):

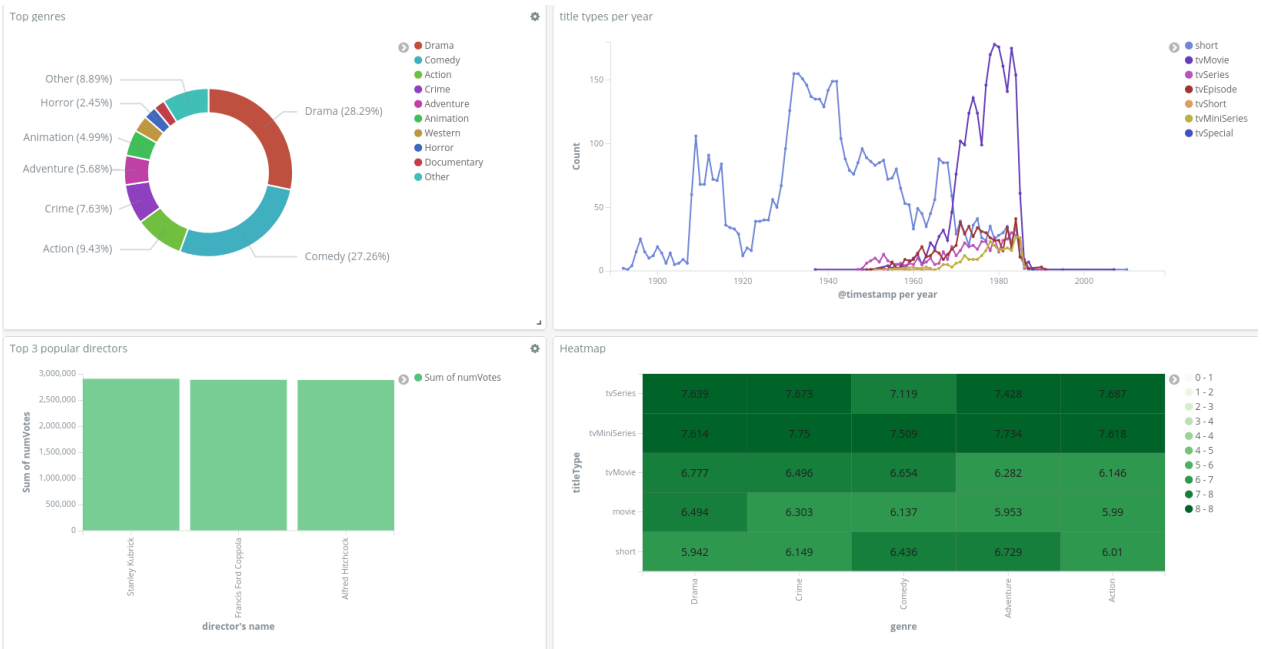


4. VIZUALIZACE – HEATMAPA

Heatmapa, kde na ose Y je 5 nejčastějších žánrů a na ose X je 5 vybraných typů titulů, zobrazuje průměrné hodnocení dané kombinace žánru a typu titulu:

DASHBOARD

Výsledný dashboard vypadá následovně (plné rozlišení viz přílohy):



ZÁVĚR

Semestrální práce byla dobrým průletem stěžejních technologií, se kterými jsme se v tomto předmětu seznámili. Největším oříškem byl výběr vhodných datasetů, jelikož jsem se chtěl vyhnout náhodnému generování. Na všechny ostatní úlohy pak stačily základní znalosti ze cvičení. Jediná věc, která mě mrzí a na kterou mi nezbyl čas, je přímé propojení HDFS a logstashe, mimo to jsem s výsledky své semestrální práce vcelku spokojený.