

Team #9

Game Popularity Prediction System

 Jiaao Yu
 Miner Yang
 Runjie Li





Outline

Goals

Use Cases

Data Source

Methodology

Scala Utility / Repository

MileStones

Acceptance Criteria

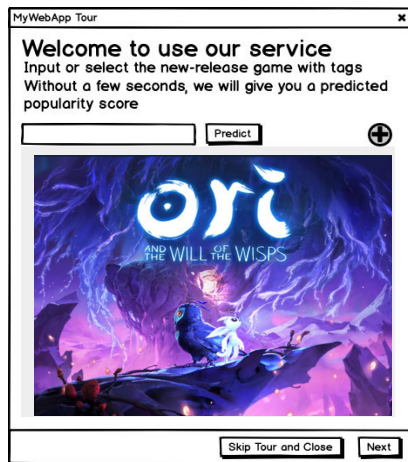
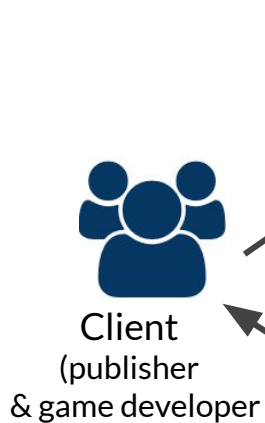
Further Expectation

Goals

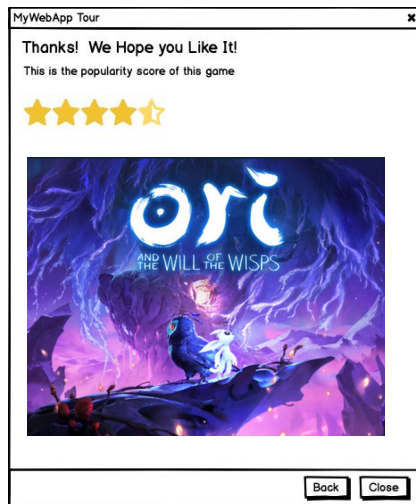
- Predict popularity of a new released game
- Back the predicted result to our client (game developer and publisher)
- Popularity would be classified into multi-level
- Assist to evaluate steam game market better



Use Cases



Front-end



Database

Write & Store

Read

Request

Send result back

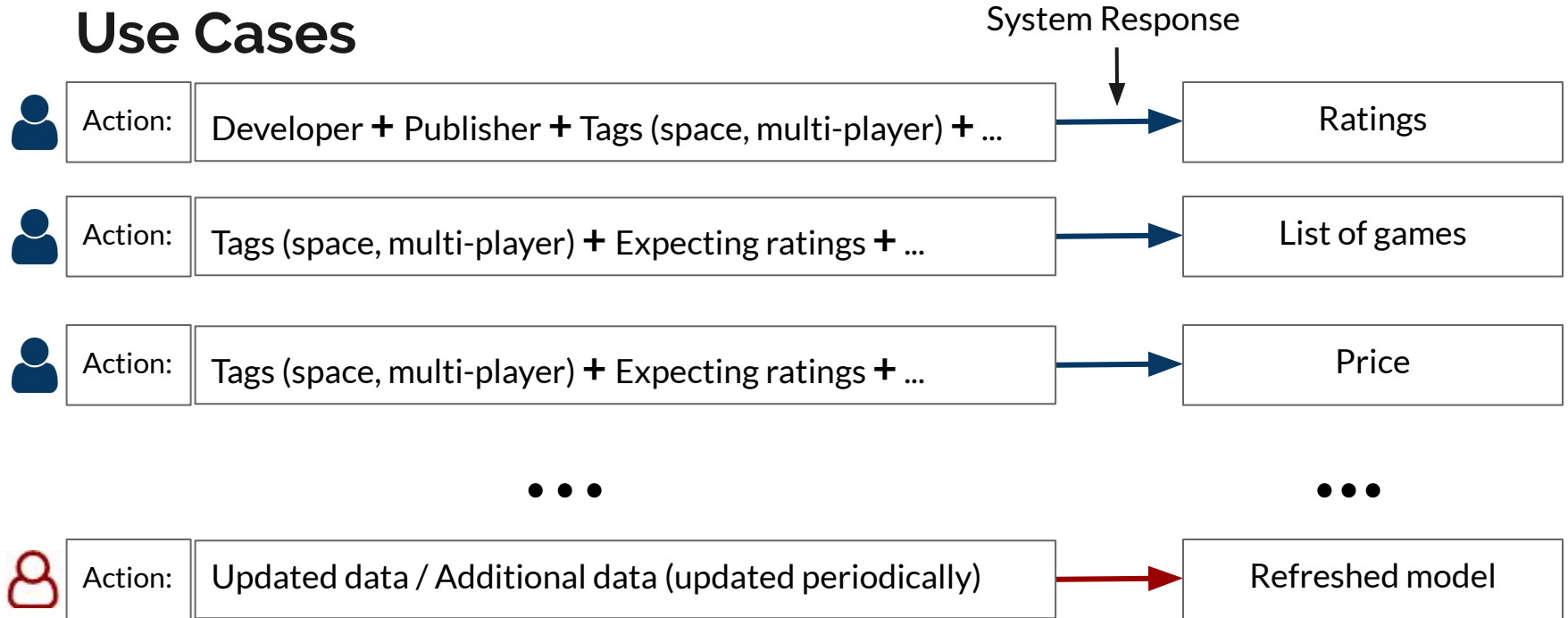


Back-end

(modeling)



Use Cases





Data

- Main Source [Steam Store Games \(Clean dataset\)](#)
- Combined Source [Steam Reviews Dataset](#)
- There are 27075 game id in the dataset

steam.csv	27075 rows	18 columns
steam_description.csv	27334 rows	4 columns
steam_media_data.csv	27332 rows	5 columns
steam_requirement.csv	27319 rows	6 columns
steam_support_info.csv	27319 rows	4 columns
steamspy_tag_data.csv	29022 rows	372 columns

Methodology

To Build a Games Rating Prediction System in Steam Store

STEPS

- **Data Ingest:** Using Spark Streaming to read from kaggle dataset website
- **Preprocessing:** Clean and break down the text of JSON format using scala
- **Feature extraction:** Analysis correlations to find suitable features
- **Modeling:** Training several models to got a qualified one (KNN, RandomForest, etc.)
- **Feedback:** Using qualified models predict result then back to user



Scala Utility

- Extract and ingest data
- Map-reduce work (if necessary)
- Process Data (filter and clean)
- Unit Test
- Use Spark MLlib to train model

Other

- Other Machine learning skills(optional)
- Using JS to build Front-End page
- Using Java/Scala to build Back-End
- Akka to handle concurrent response

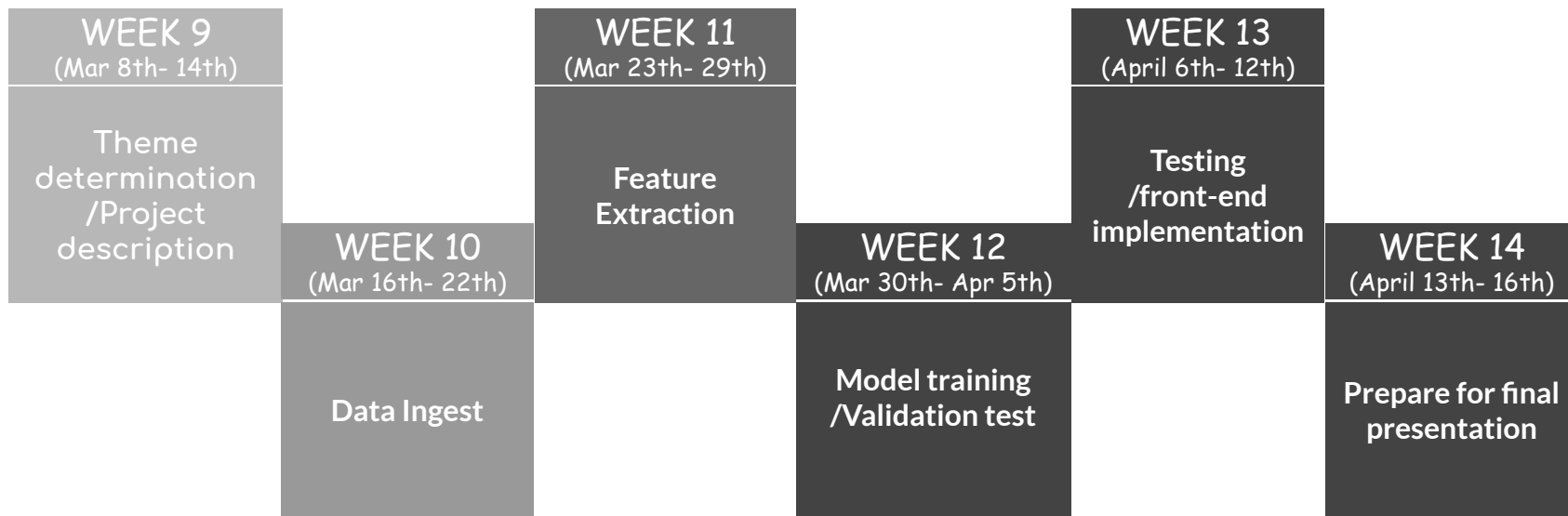



Github Repository

https://github.com/MinerYang/Team09_GamePopularity

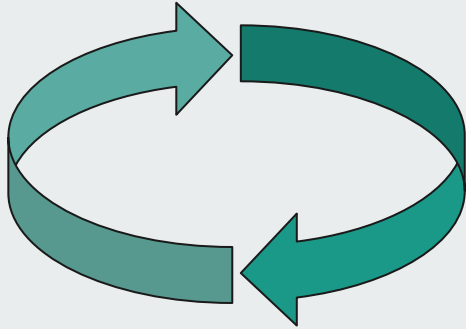
Team #9

MileStones





Acceptance Criteria



		Ture/Actual				
		1	2	3	4	5
P R E D I C T E	1					
	2					
	3					
	4					
	5					

confusion matrix for multi-class

- Average precision: > 65%
- Average recall: > 65%
- Average f1-score: > 65%
- Response time should less than 4 seconds



Further Expectation



- Expect more precise model
- Expect to use service like AWS
- Expect to build a user-friendly reactive RESTful web application
- User group could be specified:
 - Players looking for a game
 - Producers planning for a new game
 - Publishers pricing a game

Thanks!

