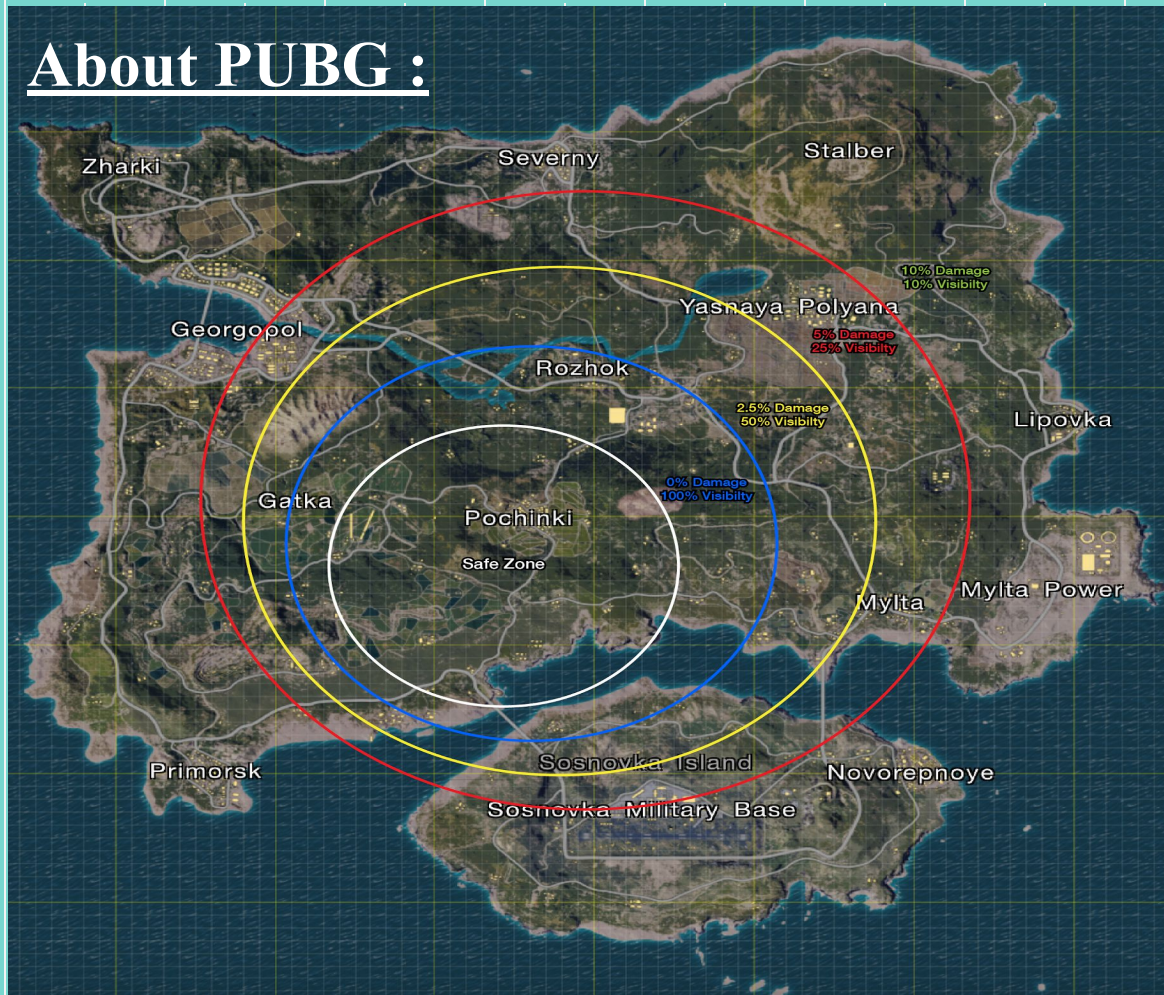




PlayerUnknown's Battlegrounds Data Analysis Using Spark

Team 6: Beiwen Guo
Zhenyu Zhu

About PUBG :



Our Goals

- Processing and visualizing matches data from the popular video game PUBG to determine if specific strategies or behaviors are related to winning the game.
- Gain more hands-on experience of coding with Scala.
- Learn to use Spark to build big data engineer project.
- With the assistance of Spark and Scala, train our skills of ingesting and processing data with cloud resources.



Use Case 1



Actor : Business User

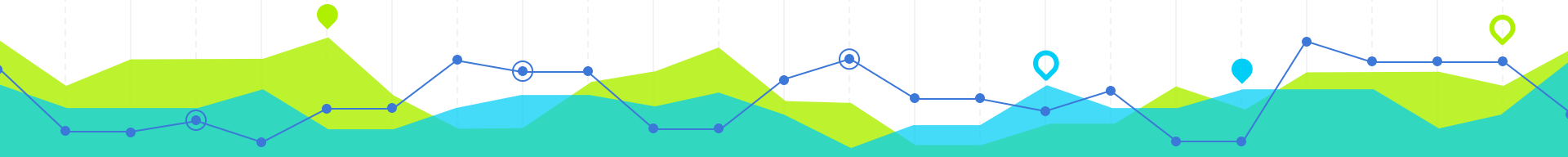
Input :

Matches data (csv format)

Output :

DataFrame with clustering results.
Plots helping understanding data.

All players will be divided into several groups according to their travel preference and battle performance. We trained k means model within dataset we got from kaggle. Models can be fastly retrieved from AWS S3 bucket.



Use Case 2



Actor : Business User

Input :

Matches data (csv format)

Output :

DataFrame with prediction indicating if such a player is a winner in a single game.

Predictions will be made by our classifier which was built based on multilayer perceptron classifier from Spark MLlib.



Methodology



Filtering data

Discard data which will not be used and ingest data into an available format.



Spark MLlib

Building and training model using Spark MLlib.



AWS

Using AWS resources to process large scale data.



Apache Zeppelin

Using Apache Zeppelin to generate visualization output.



Hortonworks Data Cloud

With assistance of HDCloud, configure computing resources we need.

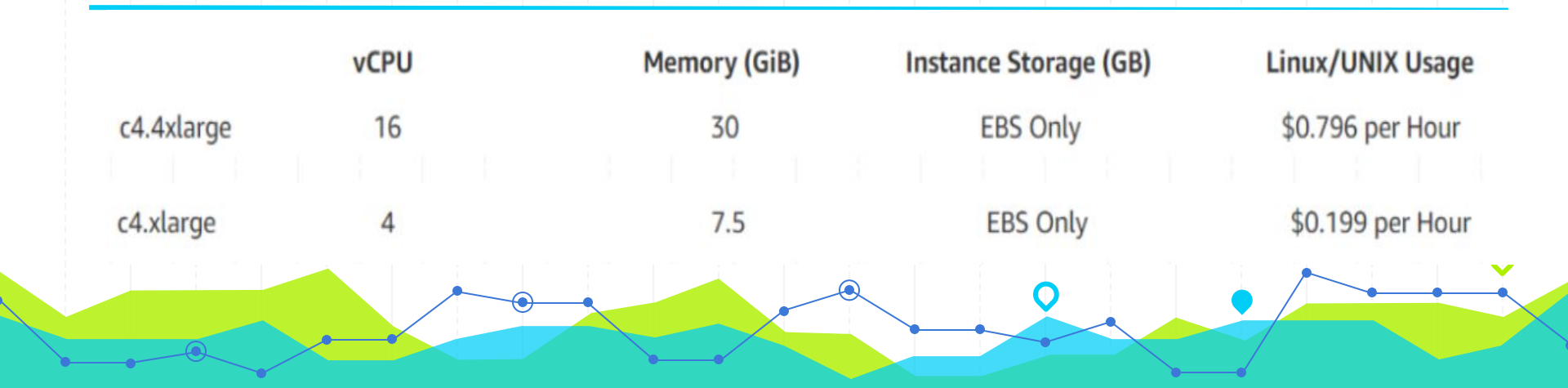


AWS & HORTONWORKS DATA CLOUD

Cloud Resource :

Computing Cluster: We deployed a spark cluster on AWS infrastructure with the assistance of HDC.

S3 Bucket: Reliable storage solution for our trained models.





Clustering Model

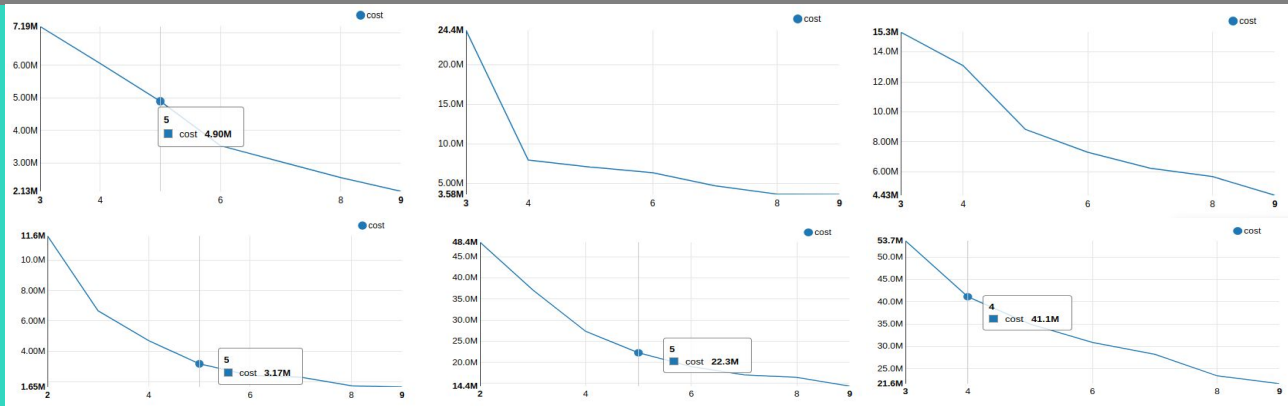
From Spark MLlib

1

K-Means

The work we have done on clustering can be divided into two parts. One is clustering according to player's travel preference, the other one is clustering according to player's battle performance. Each part is also divided into 3 subsets according to the game mode (Solo, Duo or Squad). In all, we have 6 models for clustering.

For each model we used elbow method to determine the number of k.





Classification Model

From Spark MLlib

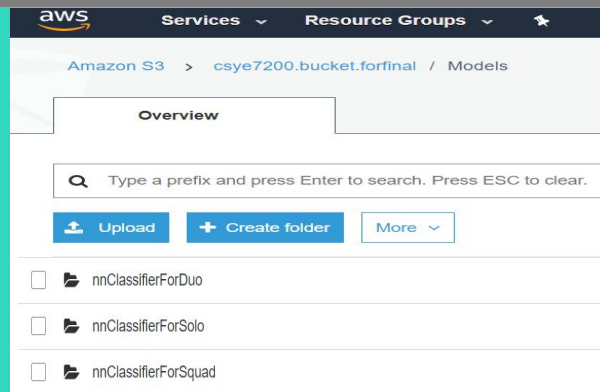
2



Multilayer perceptron classifier

Like what we did for clustering, we also separated players by game mode for classification. So we also have three models for determining if a player is a winner in a single game.

We tuned the models for several times to get the parameters we eventually used. Although we are running all this on a 4 nodes cluster, it was still a very time consuming job.



Programing In Scala

- ❑ PreliminaryProcess
- ❑ Ingest
- ❑ Player
- ❑ Clustering
- ❑ ClusteringByBattle
- ❑ ClusteringByDistance
- ❑ Classification
- ❑ Usecases
- ❑ Unit Test

▼ edu.neu.coe.CSYE7200_Team6_2018Spring

▼ Classification

○ Classification

▼ Clustering

⌈ Clustering

○ ClusteringByBattle

○ ClusteringByDistance

▼ Ingest

⌈ Ingest

⌈ Player

⌈ PreliminaryProcess

▼ Usecases

○ Usecases

▼ test

▼ scala

▼ edu.neu.coe.CSYE7200_Team6_2018Spring

▼ Classification

⌈ ClassificationSpec

▼ Clustering

⌈ ClusteringByBattleSpec

⌈ ClusteringByDistanceSpec

⌈ ClusteringSpec

▼ Ingest

⌈ IngestSpec

⌈ PlayerSpec

⌈ PreliminaryProcessSpec

Data Source :

- ★ All data are from kaggle called PUBG Match Deaths and Statistics.
There are approximately 10 million rows and 20 GB in all.

Reference:

- ★ <https://www.kaggle.com/skihikingkevin/pubg-match-deaths>



Acceptance Criteria Report

- The speed of our system to finish a visualization work should be faster than 20s/GB. 😊

18 charts took 831s for 8GB data.

$$831 / 18 / 8 = 5.77$$

The speed of our system is 5.77s / GB.

Took 7 min 44 sec. Last updated by admin at April 22 2018, 10:27:21 PM.

Took 20 sec. Last updated by admin at April 22 2018, 10:28:58 PM.

Took 1 sec. Last updated by admin at April 22 2018, 10:29:01 PM.

Took 2 min 34 sec. Last updated by admin at April 22 2018, 10:31:38 PM.

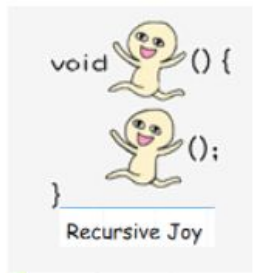
Took 3 min 12 sec. Last updated by admin at April 22 2018, 10:37:36 PM.

Acceptance Criteria Report

- Classification model can determine if the user is a winner. The accuracy should be at least 70%.

$$\text{Accuracy} = \frac{tp + tn}{tp + tn + fp + fn}$$

Test set accuracy = 0.9891577563669826
Test set accuracy = 0.9785052639146727
Test set accuracy = 0.9597469032400867



$$\text{Precision} = \frac{tp}{tp + fp}$$

res64: Double = 0.6446023989338072

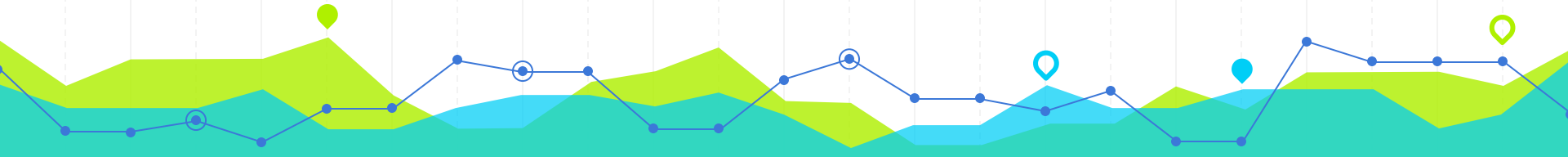
res65: Double = 0.5163511187607573

```
for(n ← Range(0, 100)) yield | 
```

[illegible]

Demo Link :

<https://ec2-34-207-189-154.compute-1.amazonaws.com/>



THANKS!

Repo Link :

https://github.com/beiwen/CSYE7200_FinalProject_Team06

