

APAN 5450 Cloud Computing

Project Implementation

School of Professional Studies, Columbia University

APAN5450_002 Cloud Computing

Group 7

Xiaoting Teng, Yiwen Yin, Chang Yuan, Keqi Yu, Ziyi Xiao

December 12th, 2022

Background and Business plan:

The group plans to provide recommendations for Youtube Channel, knowing what is trending in videos could help businesses target advertisements specific to viewer country and taste. So, the group explores the key factors that influence viewers' preferences and content based on variables such as use likes & dislikes, publish time, comment counts and views count to predict audiences behaviors.

Data Source and Description:

The dataset is a version of Youtube Trending Videos Statistics from Kaggle which includings Youtube videos that are most popular on a daily basis. The size of data is 2.79 GB Including Json files and csv across 11 regions around the world, respectively India, USA, Great Britain, Germany, Canada, France, Russia, Brazil, Mexico, South Korea, and Japan. And based on the main business plan, the team plans to use 6 regions(1.5 GB) to do analysis, respectively USA, Canada, France, Russia, South Korea and Japan.

Team Responsibilities:

Xiaoting Teng:	VPC setup and EC2 instance setup with key pairs
Yiwen Yin:	Upload and restore data in S3
Keqi Yu:	Clean data in AWS Glue
Chang Yuan:	Integrate Data from S3 and configuration
Emily Ziyi Xiao:	Classification in Sagemaker

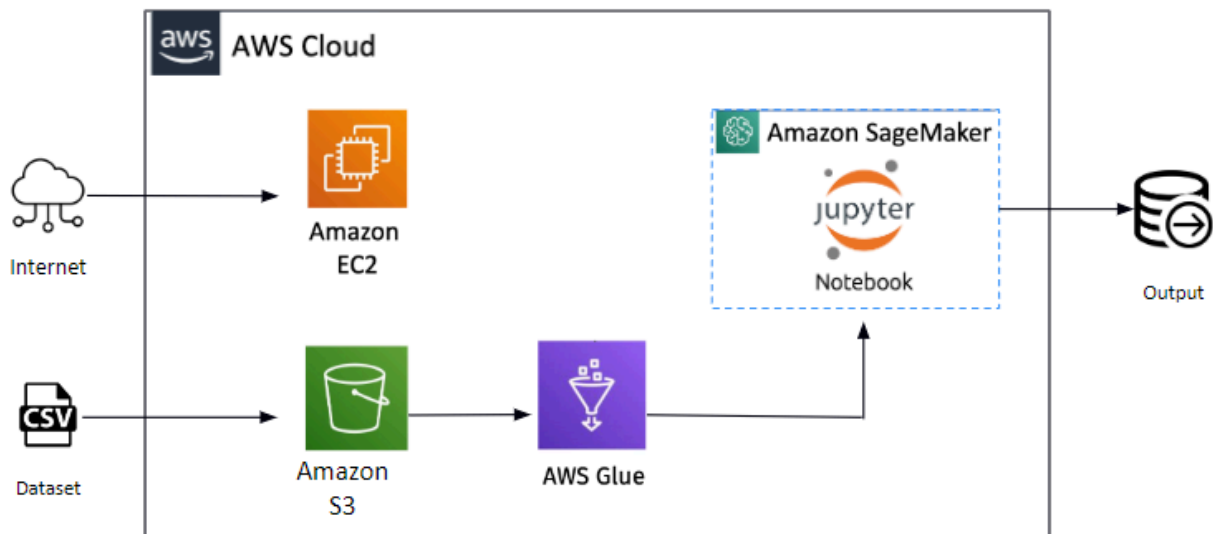
Cloud Architecture Description:

- Amazon EC2 (future use)
 - The reason for our belief that EC2 will be useful in the future is that we trained our model to solve YouTuber's needs and to be easy to use. Flasks of EC2 might be used with EC2.
- Amazon Virtual Private Cloud (VPC)
 - AWS VPC uses security groups as a firewall to control traffic at the instance level, while it also uses network access control lists as a firewall to control traffic at the subnet level. VPC provides much more granular control over security.
- Amazon Simple Storage Service (S3)
 - The group uses S3 because it can assist the team store data at the lowest cost, backup, and restore data, as well as providing great monitoring to ensure data security, S3 is essential in this research. There is also a need to set up their own securities plans such as take it own control and permission for the system.
- Amazon Glue
 - Since AWS Glue is a serverless data integration service that makes data preparation simpler, faster, and cheaper. Thus, the group uses it to help with data cleaning and visualization.

- Amazon Sagemaker
 - Prepare, build, train, and deploy high-quality machine learning models quickly by bringing together a broad set of capabilities purpose-built for machine learning, so the group used Sagemarker for machine learning to predict audience behaviors and count views.

ETL Process:

- Internet to EC2 Instance, mainly used in the future with flask
1. Upload our dataset to S3
 2. Use AWS Glue to perform data cleaning
 3. Sagemaker to generate ML model
 4. Conduct output



Security Plan:

For the security plan, the group worked on three different aspects of the security plan.

Firstly, an IAM role is an IAM identity that you can create in your account that has specific permissions. Since we cannot create the role right now, we use the default LabRole to manage the permission. In the future, when we get permission we can achieve this. **It is a web service that helps the group securely control access to AWS resources. The group uses Identity and Access Management to control who is authenticated and who has permissions to use resources.**

Secondly, the group created public subnets in the group's EC2 instance that can use various platforms and devices to access, analyze, and execute analysis on the data. There are also different availability zones for users to choose. Furthermore, the group set up SSH access in EC2 that can communicate with further technologies.

Thirdly, the NAT gateway (Network Address Translation Gateway) permitted proxy access to application servers for a group of servers in private subnets and it was able to keep security access from private subnets to the internet.

Cost Analysis:

The group used AWS price calculator to compute monthly cost of servers that are used in the project. The total cost of the usage is upfront \$29.78 and monthly with \$55.10.

For AWS VPC, the total NAT Gateway cost and data processing cost is about \$32.94 and total privatelink endpoints and data processing costs \$7.32;

For AWS EC2 instance, Amazon EC2 instance saving plan costs monthly \$2.48 and EC2 instance upfront costs \$29.78;

For AWS S3, the cloud storage costs \$0.5 monthly, and AWS Glue costs \$ 8.86, and AWS sagemaker costs \$3.

Success Criteria:

Quantitative:

- Cost-effectiveness
The cost of implementing technologies for applicable data and storage is saved for analysis, which the analysts can directly analyze by using this prepared AWS technologies
- Data Size
The data is securely and successfully imported into the Cloud ready to analyze and predict variables of interest

Qualitative:

- Data Quality
Assume raw data from data resources are up-to-date and well-collected, the processed data should be reliable, and the search result from processed data should be accurate.
- Meaningful
The analysis will provide meaning to social reality by understanding how an individual is subjectively perceived.

Implementation & Results

1. Setting up VPC and key pairs and launching the EC2 instance, and creating private and public subnets and associate each together. Created a security group that associated with the group's instance.

VPC ID  vpc-0db3769dbbe01bc25	State  Available	DNS hostnames Enabled	DNS resolution Enabled
---	--	--------------------------	---------------------------

Instance summary for i-07e2bd36bf10a0c2 (Group 7 Web Server 1)
[Info](#)

Refresh

Connect

Instance state ▼

Actions ▼

Updated less than a minute ago

Instance ID i-07e2bd36bf10a0c2 (Group 7 Web Server 1)	Public IPv4 address 52.90.154.101 open address	Private IPv4 addresses 10.0.2.177
IPv6 address -	Instance state Running	Public IPv4 DNS ec2-52-90-154-101.compute-1.amazonaws.com open address
Hostname type IP name: ip-10-0-2-177.ec2.internal	Private IP DNS name (IPv4 only) ip-10-0-2-177.ec2.internal	
Answer private resource DNS name IPv4 (A)	Instance type t2.micro	Elastic IP addresses -
Auto-assigned IP address 52.90.154.101 [Public IP]	VPC ID vpc-0db3769dbbe01bc25 (Group 7 lab-vpc) 	AWS Compute Optimizer finding Opt-in to AWS Compute Optimizer for recommendations. Learn more
IAM Role -	Subnet ID subnet-0e1d91b173e545bd5 (group-7-lab-subnet-public2) 	Auto Scaling Group name -

- Created bucket and upload dataset in S3 with subsequent access. Created a job for cleaning data in AWS Glue and AWS Sagemaker as well.

Access control list (ACL)

Edit




Grant basic read/write permissions to other AWS accounts. [Learn more](#)

This bucket has the bucket owner enforced setting applied for Object Ownership
 When **bucket owner enforced** is applied, use bucket policies to control access. [Learn more](#)

Grantee	Objects	Bucket ACL
Bucket owner (your AWS account) Canonical ID: d0911766e6081fa57f20897186b20c1d23088a2c901cac58530d525d43df4b5e	List, Write	Read, Write
Everyone (public access) Group: http://acs.amazonaws.com/groups/global/AllUsers	-	-
Authenticated users group (anyone with an AWS account) Group: http://acs.amazonaws.com/groups/global/AuthenticatedUsers	-	-
S3 log delivery group Group: http://acs.amazonaws.com/groups/s3/LogDelivery	-	-

Job name	Type	Last modified	AWS Glue version
group7dataclean	Glue ETL	12/11/2022, 10:18:33 PM	3.0

- Created domain of AWS Sagemaker with the VPC setted up for connecting notebook and using Machine Learning Model with Jupyter Notebook.

Name group7	Status  Ready	Domain ID  d-6oplofmd2joc
Created Thu Dec 08 2022 09:48:01 GMT-0500 (Eastern Standard Time)	Last modified Thu Dec 08 2022 09:52:53 GMT-0500 (Eastern Standard Time)	VPC vpc-0db3769dbbe01bc25
Authentication method AWS Identity and Access Management (IAM)	Execution role  arn:aws:iam::715923724147:role/LabRole	

Data cleaning

1. Connect to S3 bucket from AWS Glue and merge all the data files together

```
# Connect to S3 bucket
bucket = 'group7bucketnew'
file_key = ['US_youtube_trending_data.csv', 'CA_youtube_trending_data.csv', 'FR_youtube_trending_data.csv',
s3uri = []
for i in file_key:
    s3uri.append('s3://{}/{}/{}'.format(bucket, i))
```

```
data = pd.DataFrame()
for i in s3uri:
    df = pd.read_csv(i)
    df["origin"] = i[21:]
    data = data.append(df)
```

1012039 rows × 17 columns

The dataset before the data cleaning

2. Convert Data Type for Date Column to datetime format and convert datetime to date, month, year, time, hour

```
# Transforming Trending date column to datetime format
df['trendingAt'] = pd.to_datetime(df['trendingAt'], format='%Y-%m-%dT%H:%M:%SZ')
df['publishedAt'] = pd.to_datetime(df['publishedAt'], format='%Y-%m-%dT%H:%M:%SZ')
```

```
df.insert(loc=3, column='published_date', value=df.publishedAt.dt.date)
df.insert(loc=4, column='published_month', value=df.publishedAt.dt.month_name())
df.insert(loc=5, column='published_day', value=df.publishedAt.dt.day_name())

df.insert(loc=10, column='trending_date', value=df.trendingAt.dt.date)
df.insert(loc=11, column='trending_month', value=df.trendingAt.dt.month_name())
df.insert(loc=12, column='trending_day', value=df.trendingAt.dt.day_name())
```

```
df.insert(loc=6, column='published_time', value=df.publishedAt.dt.time)
df.insert(loc=7, column='published_hour', value=df.publishedAt.dt.hour)

df.insert(loc=13, column='trending_time', value=df.trendingAt.dt.time)
df.insert(loc=14, column='trending_hour', value=df.trendingAt.dt.hour)
```

3. Delete abnormal data

```
mask = (data.view_count<=0)
df = data.loc[~mask]
```

Finally, after the data cleaning, the cleaned dataset has 1011848 rows \times 27 columns

Implement Results

Most Viewed, liked and Disliked Videos on Trending Page

	view_count	likes	dislikes	comment_count
title				
C'est trop long ! - Détournement par @Daadhoo (instagram)	39949.0	4991.0	27.0	218.0
MORGENSHTERN & Тимати - EL Система (ПАРОДИЯ El Problema)	38214.0	6012.0	109.0	407.0
pitié que ça s'arrête !	1107079.0	120180.0	0.0	1659.0
В Кремле пылает — все мы видели эти билборды! Беда наша — глупость и цинизм — Марунич	279604.0	3404.0	0.0	570.0
Зеленский ты трус!! Одесса ВСТАЛА ПРОТИВ ЗЕ !	123885.0	7822.0	325.0	1372.0

We develop a function to product the bar plot

```
def plot_barplot(x, y,
                 xticks=None,
                 yticks=None,
                 xlabel=None,
                 ylabel=None,
                 title=None,
                 figsize=(15,8),
                 rotation=0,
                 barh=False):

    fig, ax = plt.subplots(figsize=figsize)

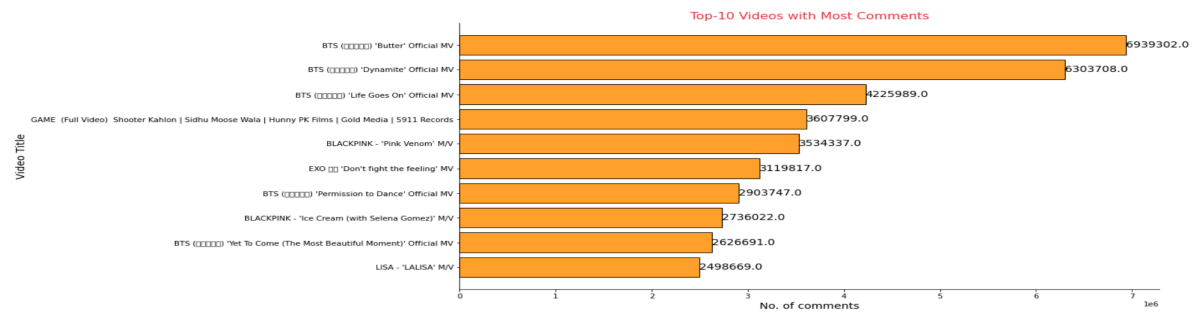
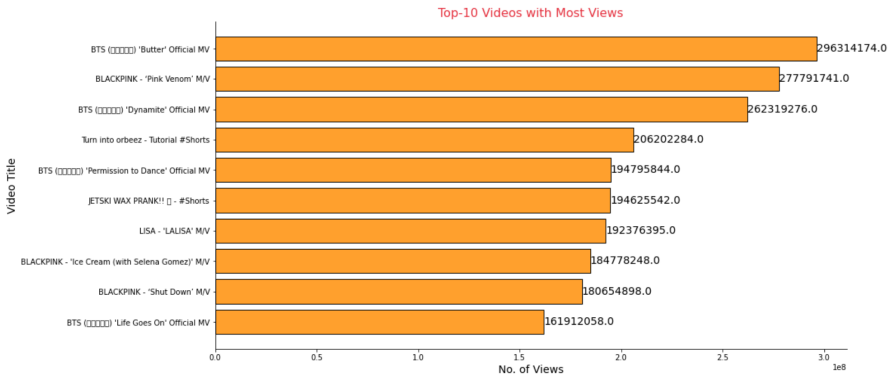
    if barh:
        g = plt.barh(y=y, width=x, color='orange', ec='k')
        plt.yticks(ticks=y, labels=yticks)

        for rect in g.get_children():
            h = rect.get_height()
            w = rect.get_width()
            x = rect.get_x()
            y = rect.get_y()
            plt.annotate(f"{w}", (w, y+h/2), ha='left', va='center', fontsize=14)
    else:
        g = plt.bar(x=x, height=y, color='orange', ec='k')
        plt.xticks(ticks=x, labels=xticks, rotation=rotation)

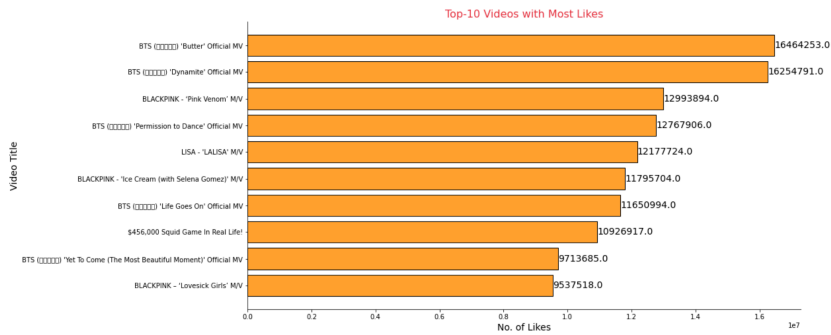
        for rect in g.get_children():
            h = rect.get_height()
            w = rect.get_width()
            x = rect.get_x()
            plt.annotate(f"{h}", (x+w/2, h), ha='center', va='bottom', fontsize=14)

    plt.xlabel(xlabel, fontsize=14)
    plt.ylabel(ylabel, fontsize=14)
    plt.title(title, fontsize=16, color="#E43D40")
    ax.spines['top'].set_visible(False)
    ax.spines['right'].set_visible(False)

    plt.show()
```

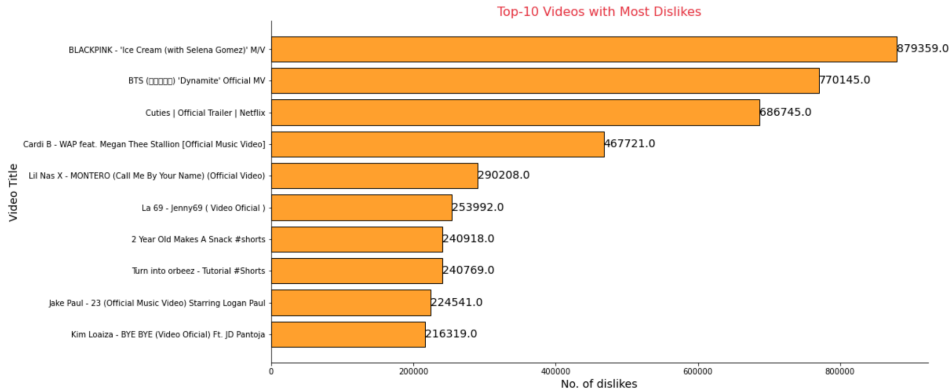


BTS and BlackPink videos make the most of the top 10 videos with most views and most comments

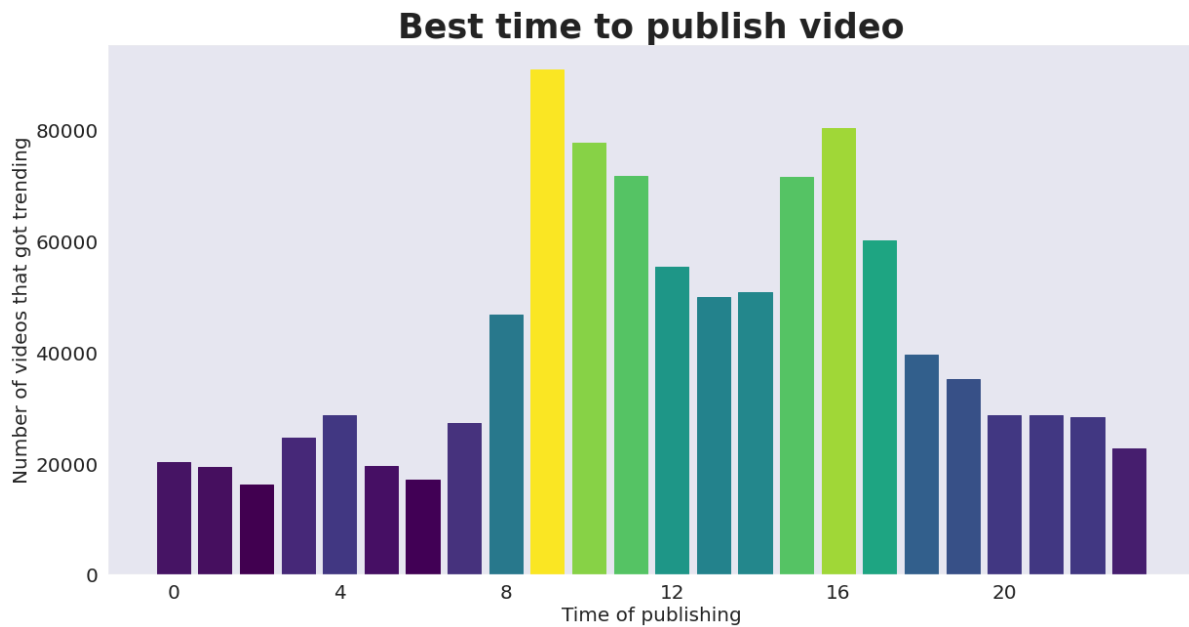


BTS and BlackPink videos make the most of the top 10 videos with most likes, which coincides with the previous plot

The video with most dislikes were also from the music video.



This graph can help YouTubers to pick the best time to publish their videos.



The graph shows the number of videos that got trending vs time of publishing, the lighter the color, the more number of count. We can see that the best range to publish video would be from 8AM to 7PM. Within the range, the top hour would be 9 AM.

We used machine learning under sageMaker for our last part of implementation. Our y variable is view_count, which represents the number that a video gets played. We used most of the other variables as our X variable to predict it (showing below).

```
columns = ['liked',
           'ratings',
           'comments_disabled',
           'comment_count',
           'ratings_disabled',
           'publishedAt_month',
           'publishedAt_day',
           'publishedAt_hour'] + common_words_1
```

We split our data into two sets, 0.8 for training and 0.2 for testing.

```
from sklearn.model_selection import train_test_split
train, test = train_test_split(
    df, test_size=0.20, random_state=42)
```

The models that we select are Support Vector Machine classification and Logistic Regression classification. The reason why we choose these two models is because they have relatively simple algorithms. We are unable to use models that are too complex due to the fact that our fund is limited.

```
from sklearn.linear_model import LogisticRegression

clf = LogisticRegression(C=1, random_state=0, class_weight='balanced', solver='newton-cg')
clf.fit(X_train, y_train)

LogisticRegression(C=1, class_weight='balanced', random_state=0,
                    solver='newton-cg')

from sklearn.metrics import mean_squared_error
y_predict = clf.predict(X_test)
RMSE_test = np.sqrt(mean_squared_error(y_predict, y_test))
RMSE_test
```

```
from sklearn.svm import LinearSVC

svc = LinearSVC(class_weight='balanced', loss="squared_hinge", penalty="l2")
svc.fit(X_train, y_train)

/opt/anaconda3/lib/python3.8/site-packages/sklearn/svm/_base.py:985: ConvergenceWarning: Liblinear failed to converge, increase "
LinearSVC(class_weight='balanced')

y_predict = svc.predict(X_test)
RMSE_test = np.sqrt(mean_squared_error(y_predict, y_test))
RMSE_test
```

We calculated the mean squared error (RMSE) of both models. After normalizing the RMSE value, we found that SVM got a slightly better result which is 0.1906, and LR got 0.2527.

```
modify_rmse
```

```
Out[56]: 0.25269208826460066
```

```
modify_rmse
```

```
Out[66]: 0.1906378710404432
```

Using our model, YouTubers can predict their View Counts for their videos and know how many audiences would view their videos.