2 DATASET

2.1 Data information

Dataset which used in this project based on **[1]** is obtained from the **Kuaishou App**, a well-known short-video platform with a lot of of subscribers. Users may view a range of short-form videos from different categories such as dance, entertainment, and fitness/sports on this site. The videos are ordered by suggesting streaming, our dataset here is the first dataset generated from real-world recommendation logs with all users' preferences on items known and also it's **not highly sparse dataset**, this data is **a fully-observed dataset** with millions of users' interaction, the data collected here by the users themselves, every single user watch a video and left a feedback which means that we haven't **almost no missing values** here.

2.2 Files structure

- **KuaiRec**
  - **data**
    - **big_matrix.csv**
    - **small_matrix.csv**
    - **social_network.csv**
    - **item_feat.csv**

You may want to note that there's a common feature between **social_network.csv** and the two main datasets which is **user_id** feature and we can join between these two dataset using this key to recommend the video to users' friends also.

You may want to note that there's a common feature between item_feat.csv and the two main datasets which is video_id feature and we can join between these two dataset using this key to get insights about the categories that's user likes.

2.3 Data Prepossessing

1. **Handle missing values**
   Check for missing values in every dataset, every dataset has no missing values except for small_matrixċsv, it has missing value in three features **(timestamp, date, time)**, it has **3.8%** missings values in these features, I'll drop these features in **big_matrix.csv** (training data), in **small_matrix.csv** (evaluation data) since we don't use these data in our predictive model.
2. **Drop duplicated values**
   Check for consistent duplicated values in every dataset, every dataset has no duplicated values except for **big_matrix.csv**, it has about one million records, duplicated in even **timestamp** feature which is totally wrong, so I'll drop them.
3. **Handling outliers**
   Check for outliers in numerical features in all datasets except for **user_id** and **video_id**, and they are **video_duration**, **play_duration)** and **watch_ratio**, outliers values in these features are **true outliers**, it means they have informative values and these values didn't become outliers due to an entry error, so I'll just normalize their values between **0** and **1**.

2.4 Basic Statistics of the Dataset

2.4.1 big_matrix.csv statistics

	user_id	video_id	play_duration	video_duration	date	timestamp	watch_ratio
count	12530810	12530810	12530810	12530810	12530810	12530810	12530810
mean	3574.377	5058.597	9027.027	1462.157	20200800	1596799000	0.9445059
std	2067.008	3090.082e	15473.43	19834.74	50.80192	1514698	1.674601
min	0.0	0.0	0.0	140.0	20200700	1592872000	0
25%	1788.0	2388.0	4218.000	7434.0	2.0200800	1596339000.0	0.3148246
50%	3578.0	4823.0	7277.0	9636.0	2.0200810	1596669000	0.7234710
75%	5343.750	7601.0	1035.0	1217.900	20200830	1.598502000	1.177644
max	7175.0	10728.0	999639.0	315072.0	20200900	1599694000	573.4571

Table 1. big_matrix.csv statistics

2.4.2 small_matrix.csv dataset statistics

	user_id	video_id	play_duration	video_duration	date	timestamp	watch_ratio
count	4676570	4676570	4676570	4676570	4494578	4494578	4676570
mean	3631.649	4975.787	8612.637	14486.45e	20200770	1596241000	0.9070695
std	2043.873	3064.837	12236.61	20467.11	48.95180	1254444	1362324e
min	14.0	103.0000	0.0	3067.000	20200700	1593801000	0.0
25%	1834.0	2370.0	5811.000	7523.0	20200720	1595210000	0.4675769
50%	3687.0	4693.0	7549.0	9600.0	20200800	1596224000	0.7691666e
75%	5421.0	7475.0	9880.0	11934.0	20200810	1597121000	1.120590
max	7162.000	10596.00	7988155	315072.0	20200900	1599321000	571.5214

Table 2. small_matrix.csv dataset statistics

2.4.3 social_network.csv and item_feat.csv dataset statistics

Note: in these dataset statistics we don't care about the mean and the median since the these statistics doesn't important and suitable for information like ID, and there are friend list and feat feature and because it's a list datatype so I considered the length of each list and I construct another feature called friend_list_len for social_network.csv dataset and feat_len for item_feat.csv.

	user_id	friend_list_len
count	472.0	472.0
min	18.0	1.0
25%	1648.0	1.0
50%	3268.0	1.0
75%	5233.5	2.0
max	7174.0	5.0

Table 3. social_network.csv statistics

	video_id	feat_len
count	10729.0	10729.0
min	0.0	1.0
25%	2682.0	1.0
50%	5364.0	1.0
75%	8046.0	1.0
max	10728.0	4.0

Table 4. item_feat.csv dataset statistics

3 ANSWERS TO THE RESEARCH QUESTIONS

3.1 Answer to the first question:

Approach: I tend to answer this question by creating a neural network model to predict the values of **watch_ratio** using **video_duration** and **play_duration** as my input features and this model will be a **regression model**, and I used **mean square error** and **mean absolute error** as my **evaluation metrics**.

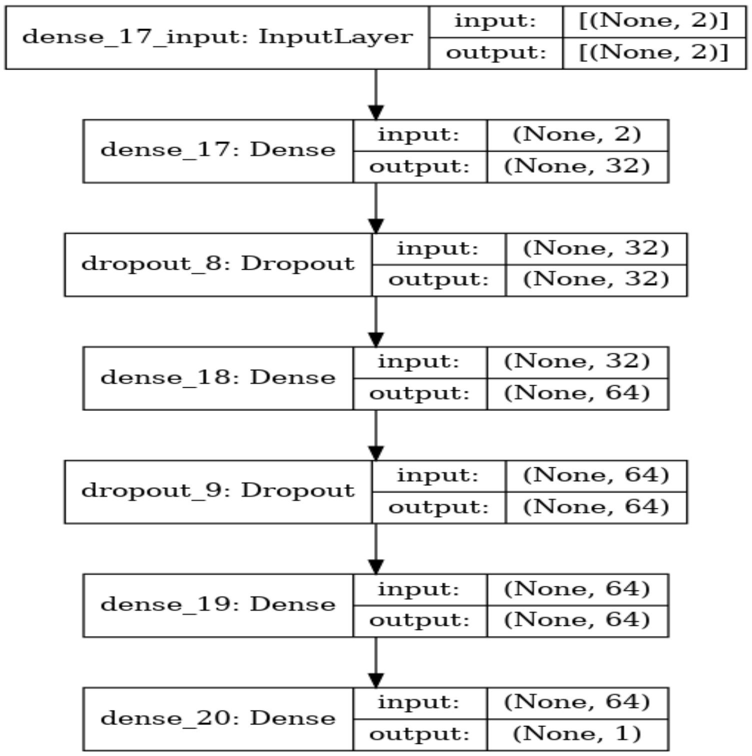

Figure 1. Neural network model to predict watch_ratio value

Main findings: After training my model for **10 epochs** and **128 batch sizes** and with **learning rate: 0.01** with **Adam optimizer** and took **20%** of my training data as **validation data**, my results was as follows.

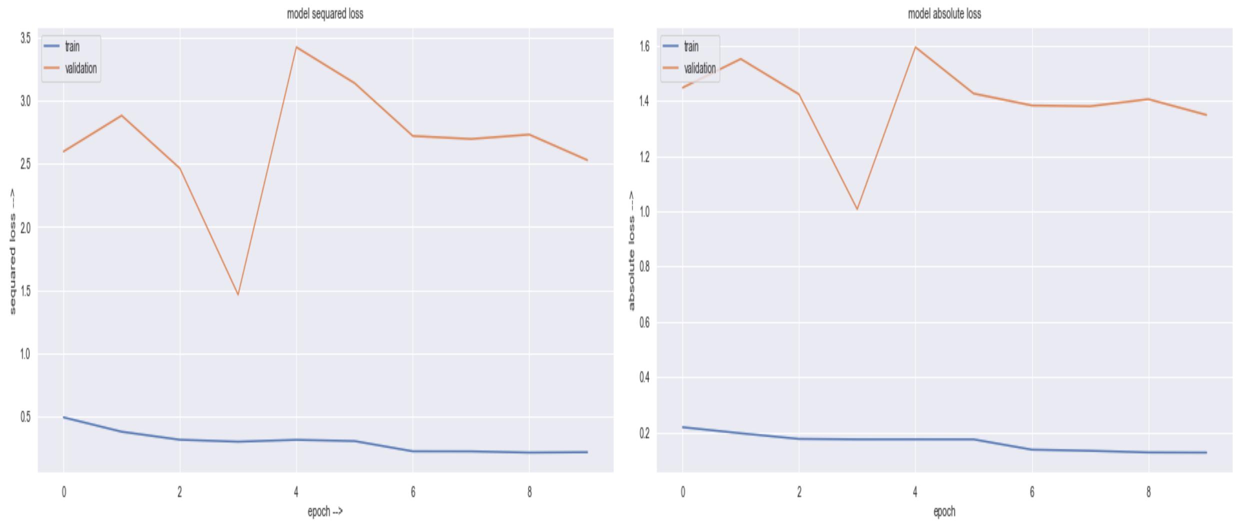

Figure 2. model squared error (left) vs model absolute error (right)

whereas the minimum mean squared error value was **0.2110** and the minimum mean absolute error was **0.1274** which is reasonable results in just 10 epochs.

After that I tested that model on input feature (video_duration and play_duration) on small_matrix.csv and I got squared loss: 1.783, absolute loss: 0.741

3.2 Answer to the second question:

I tend to answer this question by creating a more complex neural network model to classify and know whether the user liked this video or not.

I created liked feature based on watch_ratio model, if the user like that video or not based on watch_ration values, if the watch_ratio value was higher than 0.9 (watched most of the video) the the liked value will be 1.0 else will be 0.0.

I'll try to predict **liked** using **video_duration** and **play_duration** as my input features and this model will be a **classification model**, and I used **Accuracy**, **precision** and **recall** as my **evaluation metrics**.

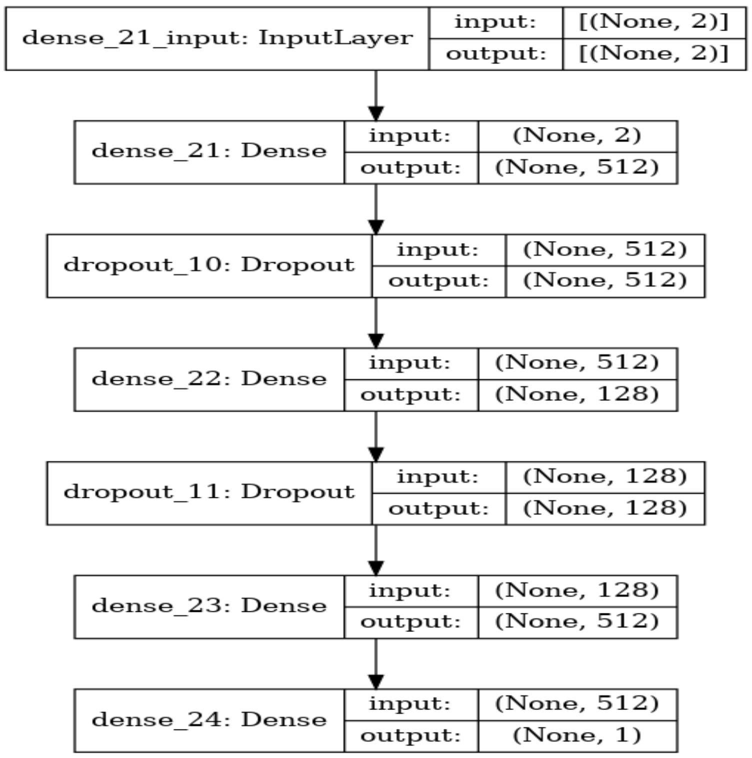

Figure 3. Neural network model to predict liked value

Main findings: After training my model for **10 epochs** and **128 batch sizes** and with **learning rate: 0.01** with **Adam optimizer** and took **20%** of my training data as **validation data**, my results was as follows.

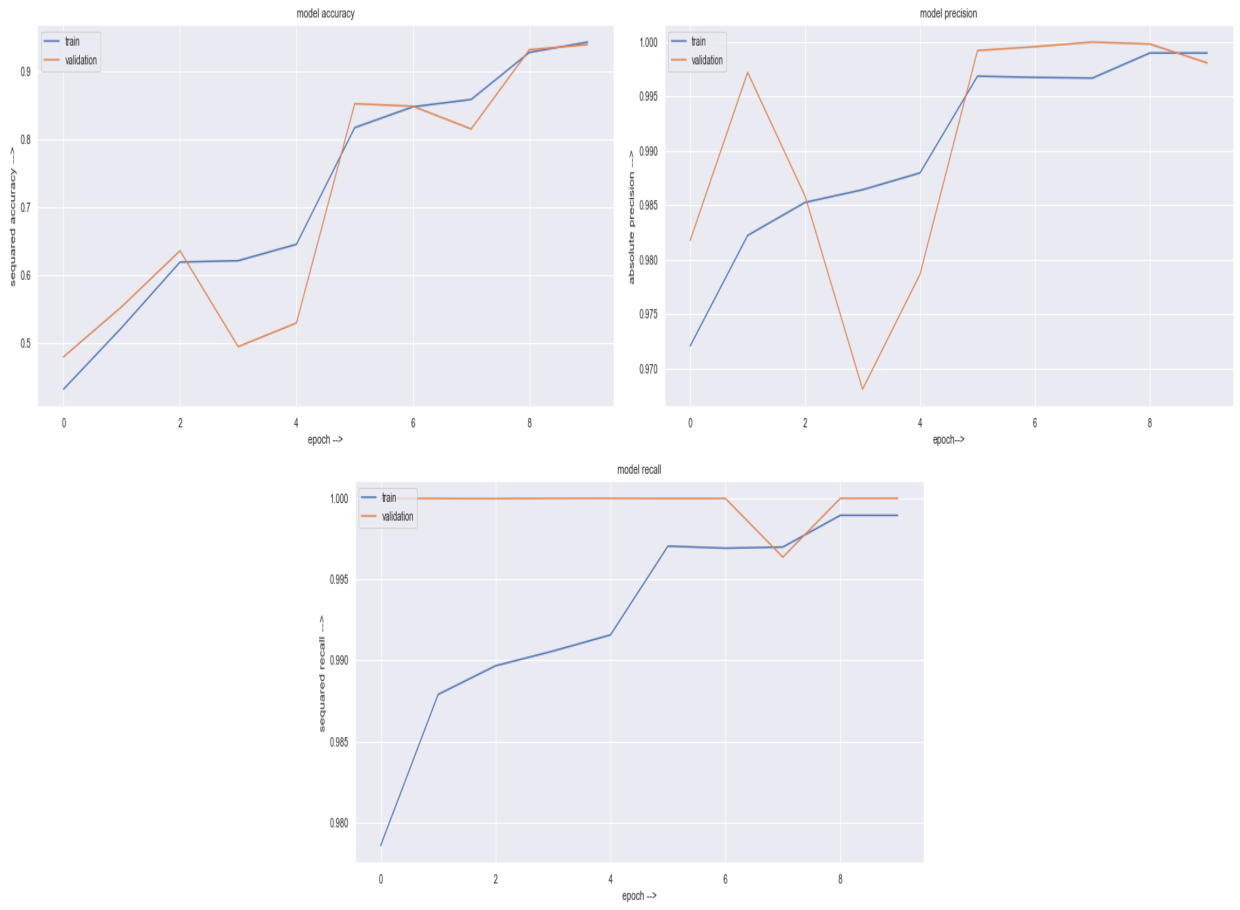

Figure 4. model accuracy (left) vs model precision (middle) vs model recall (right)

After I test this model on small_matrix.csv I got these results,Accuracy: 0.6124633550643921,recall: 0.030348394066095352, precision: 0.9926343560218811

and I think that recall is less than precision because the huge false negatives and my dataset is unbalanced.

Big note: The best models to above questions is to use neural network models, if I tried to use any machine learning model, the memory will crash since they are a huge data, so normal machine learning approaches can't answer these question unless I used big data techniques.

3.3 Answer to the third question:

we will answer this question Using statistical hypotheses test,

Approach: I'll take watch_ratio feature from big_matrix.csv and small_matrix.csv and I'll divide that feature from both datasets into two groups the first group has the watch_ratio values which is higher than specific values {about watch the full video} and less than this specific values and I'll do my hypotheses test on these groups

Null hypotheses: The shorter videos are more favorite than (more than) longer videos for the users.
alternative hypotheses: Both type of these videos have the same degree of preferences?

Main findings:

after applying our hypotheses test: we found that the value of **significance level** in **small_matrix.csv** and **big_matrix.csv** is near to **0** and we sit a condition if the **significance level** was greater than **0.05** then we accept the **null hypotheses** else we accept the **alternative hypotheses**, and in the both cases we accept the **alternative hypotheses**, and we see that's reasonable output.

4 LIMITATIONS

1. Huge dataset:

   our dataset contains 12530806 rows for training dataset, and 4676570 rows for testing dataset which consider a big data and that takes a long time for showing it not to mention preprocessing its data or try to train the train dataset or predict column values in the test dataset, and it would take a long of time if we tried to apply Collaborative Filtering techniques.

2. Cold start:

   This issue arises when new users or new items are added to the system; a new item cannot be recommended to users when it is introduced to the recommendation system without any likes or reviews, making it difficult to predict the choice or interest of users, resulting in less accurate recommendations.

3. Sparsity:

   It occurs frequently when the majority of users do not provide likes or reviews for the things they watch, causing the rating model to become highly sparse, perhaps leading to data sparsity issues. This reduces the chances of discovering a group of users with comparable ratings or interests.

4. Scalability:

   The scalability of algorithms with real-world datasets under the recommendation system is a major challenge. A large amount of changing data is created by user-item interactions in the form of ratings and reviews, and hence scalability is a major worry for these datasets.Recommendation systems inefficiently interpret findings from huge datasets; certain advanced large-scaled algorithms are necessary to address this issue.

5 TAKE-AWAY MESSAGES

So At the first we start to explore our four dataset and what's the relationship between them and how can we leverage from their data, then we do some data quality check and some data preprocessing, then we do some data analysis to explore our datasets and how their data looks likes in graphs,

Then we did some feature engineering process like dropping some columns that we won't need it to answer our questions beside that they are having null values in a specific dataset (small_matrix.csv) and they was (time, timestamp , data), then we create an output feature called liked which specify if the user like that video or not based on watch_ration values, if the watch_ratio value was higher than 0.9 (watched most of the video) the the liked value will be 1.0 else will be 0.0,

also I created a feature in the supporter dataset (social_network.csv, item_feat.csv), in the social_network.csv that feature made us know how many friends that user have and in the item_feat.csv it made us know how many tags that video have.

Then we started to answer to the first predictive question, by implementing a regression neural network model and train it on our training dataset (big_matrix.csv) and then test it on (small_matrix.csv) to predict watch_ratio values
Then we started to answer to the first predictive question, by implementing a regression neural network model and train it on our training dataset (big_matrix.csv) and then test it on (small_matrix.csv) to know wether the user liked video or not.
And For the third question and I tried to do a hypotheses test to know if the the shorter videos is better than longer videos or both these types of videos have the same preferences, and I got the the these types of videos have the same preferences and that was a reasonable results.

My future implications of this project are to predict what's the most videos the users gonna love, having this recommendation system in any video platform will increase the revenue in huge, due to the long time that will users spend on the app and the number of advertisements watchs that will increase.