Hackathon 2 – Group 3
Predict the Segment – Hotstar
Github Profile - https://github.com/swapcy/hackathon_hotstar

SWAPNIL CHAUDHARI

KISHAN ENJAPURI

NEEHA KHALFAY

CALVIN CASTELINO

Agenda

- Problem Statement
- Data Sources and EDA
- Feature Engineering
- Machine Learning Model
- Conclusion and Recommendations.

Problem Statement

- In the advertising domain, to determine the demographics of customers is a key challenge. To target the right advertisement we need to identify a customers content segment.
- Goal – Implement a Machine Learning Solution to identify watch patterns in specific customer segments.
- Problem Analysis – Classify the Interest Segment based on segment target variable. Pos = 1 and Neg = 0.

Data Source

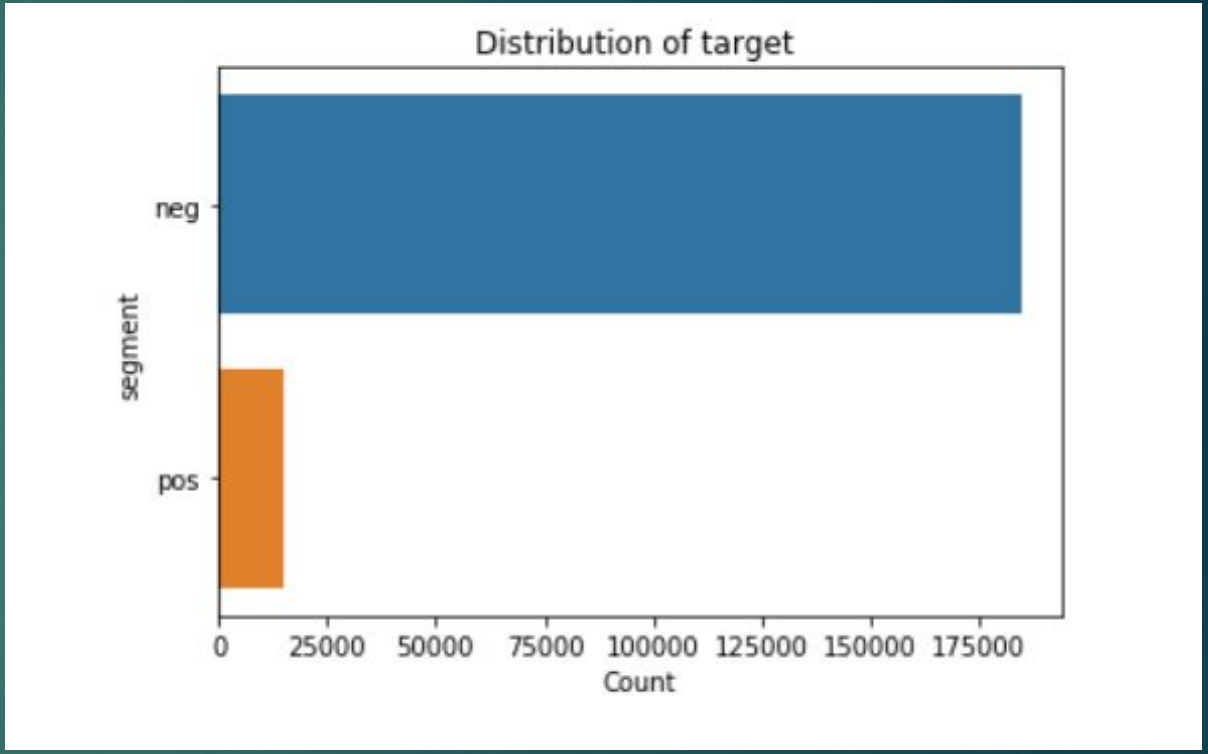
- Data Available at: https://drive.google.com/file/d/1YPVPife9QCCQPBcT5yOYi-gCpV6NABMR/view?usp=sharing
- Data Consists of:
  - 6 Features
  - Training Data Set: 200,000 customers
  - Test Data Set: 100,000 customers
  - Description ID considered as Index.

Feature Description:

- Titles – "title : watch_time" Example: "JOLLY LLB:23" (Note: Watch Time is in secs)
- Genres – "genre : watch_time" Example: "Cricket:82379", "Action:963"
- Cities – "cities : watch_time" Example: "Gurgaon:55494"
- Tod (24 hours format) – "time_of_day : watch_time" Example: "1:454", "17:5444"
- Dow (7 days format) – "day_of_week : watch_time" Example: "1:454", "6:5444"
- Segment Target Variable – Pos:1 and Neg:0 Example: "To target the right advertisement in a particular customer watch segment".

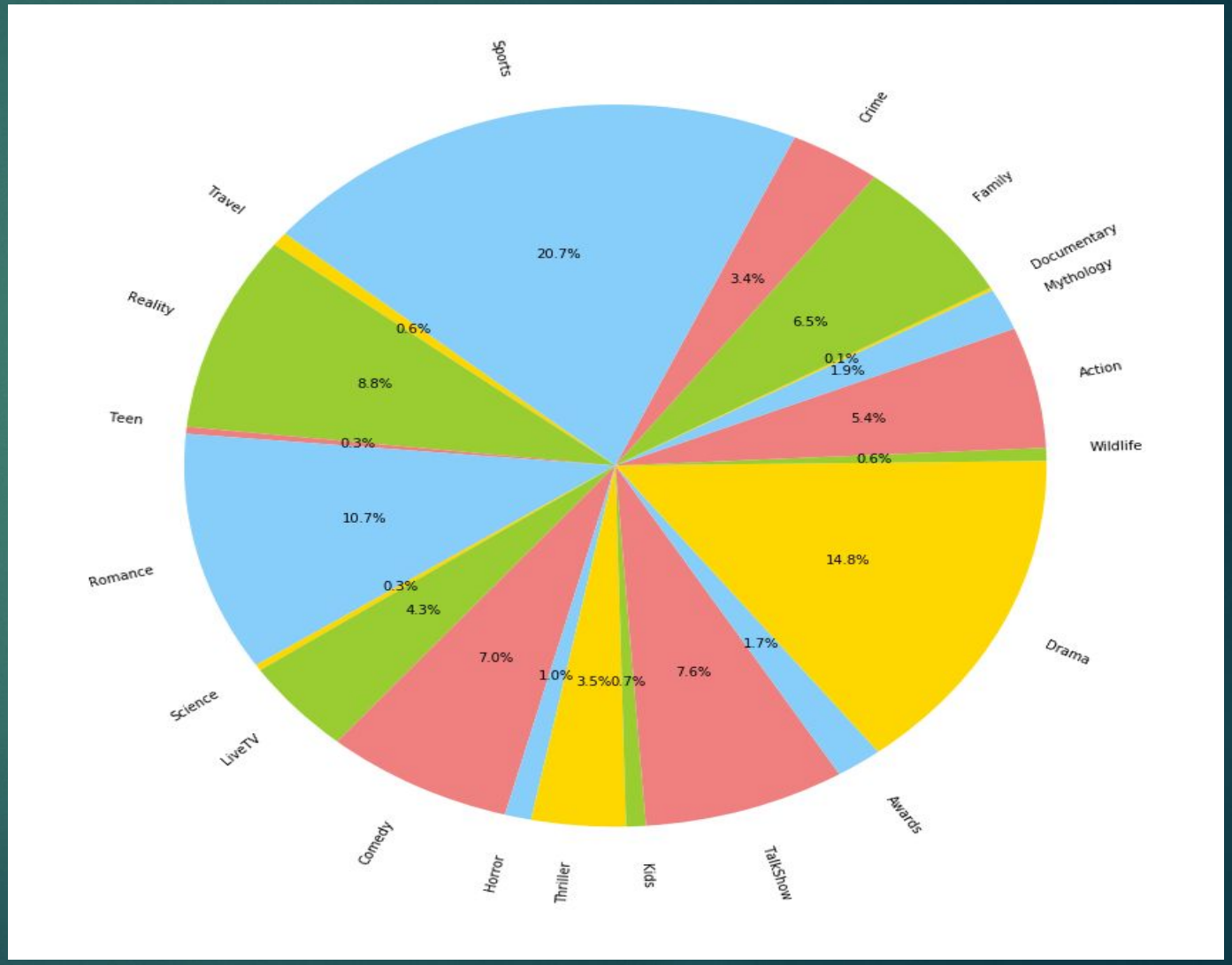
Segment Target Variable.

- To target the right advertisement in a particular customer watch segment.
- Encode Target Variable Count:
  - Neg: 0 – {Count – 184,745}
  - Pos:1 – {Count – 15,254}



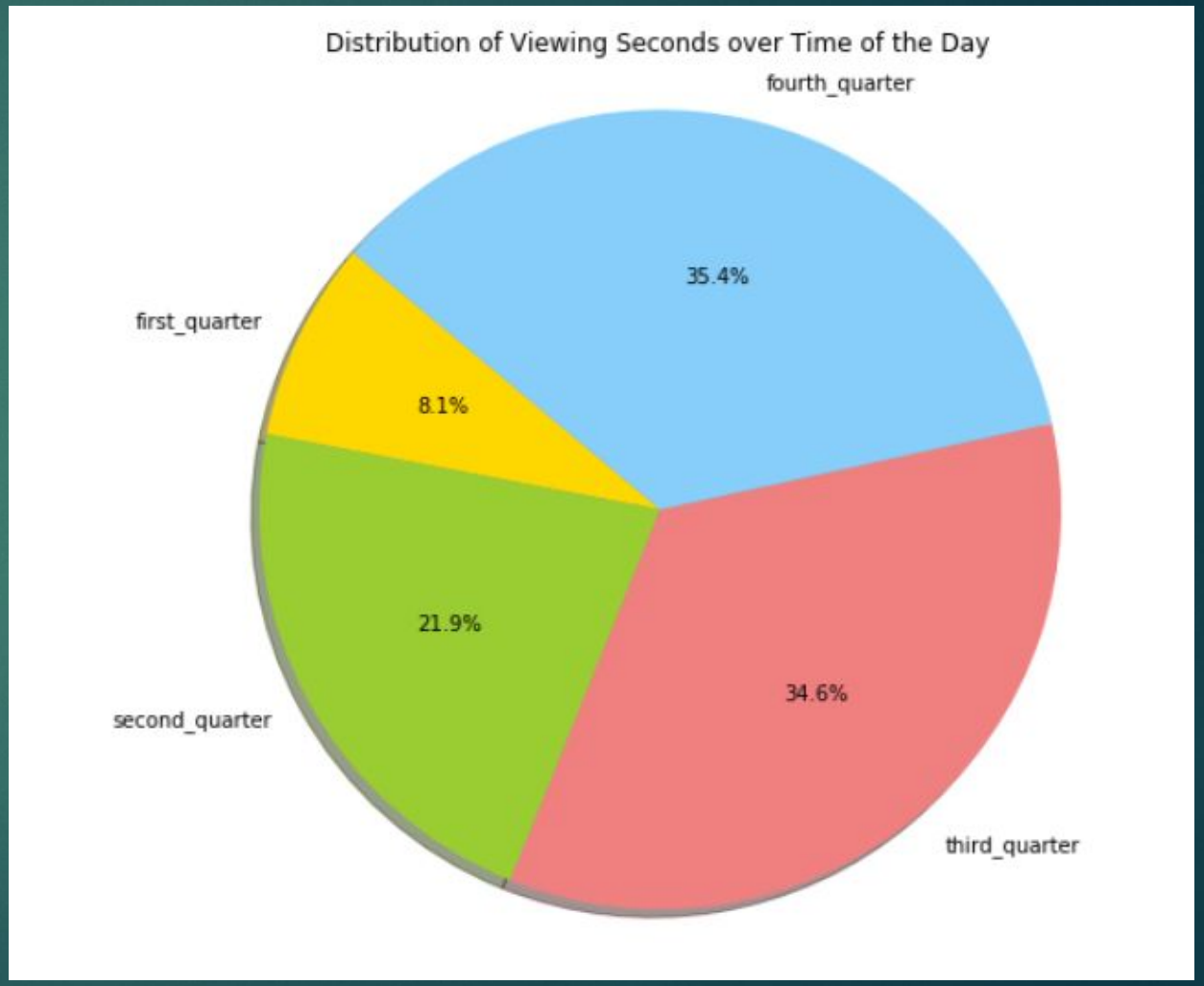
Genres -

- Genres have been considered given that customers watch pattern based on specific genres. Advertisements could be targeted in specific genre segments as below:
- Sports – 20.7%
- Drama – 14.8%
- Romance – 10.7%



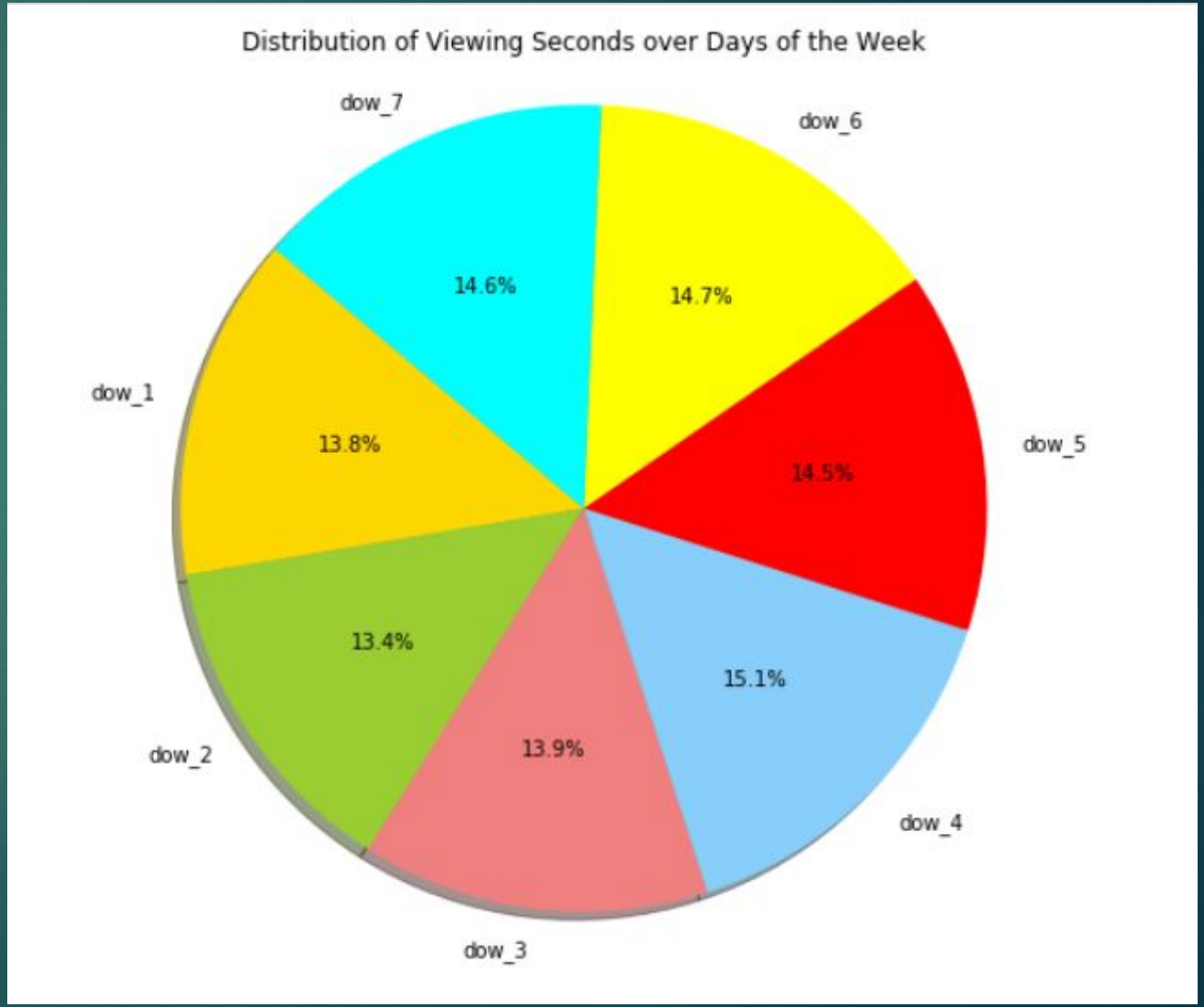
Tod (24 hours format) – time of the day

- Divided based on quarterly intervals to identify count of customer watch patterns in these intervals.
- first_quarter (1:00 am to 7:00 am) – 8.1%
- Second_quarter (7:00 am to 1:00 pm) – 21.9%
- Third_quarter (1:00 pm to 7:00 pm) – 34.6%
- Fourth_quarter (7:00 pm to 1:00am) – 35.4%



dow (7 day format) – day of the week

- Day of the week have been considered to identify particular days where there is increase in customer watch patterns.
- dow_4 – 15.1%
- dow_6 – 14.7%
- dow_7 – 14.6%



Key Findings

- Features Considered:
  - Genres, Titles, dow, tod.
- Zero NaN and No Missing Values.
- More EDA based on data and selective features.

Feature Engineering

Raw Data

- Each row represents a segment of customers with their watch patterns.
- Segment – A customer from delhi city watched cricket on the 3rd day of the week during a given time of the day between 1:00 pm to 7:00 pm.

```
In [121]: ht_dt.head()
```

Out[121]:

	cities	dow	genres	segment	titles	
train-1	gurgaon:55494,delhi:31892	1:3412,3:15878,2:1737,5:10975,4:20974,7:17820,...	Cricket:82379,Kabaddi:255,Reality:4751	neg	Top Raids: Haryana vs Services SCB:103,Day 4: ...	10:2
train-10	delhi:5862,nagar:8916,mumbai:1593	1:5745,3:3025,2:3346,5:123,4:3007,7:1108,6:10	Cricket:15640,Wildlife:730	neg	Dhoni Quits Captaincy:148,Day 4: India Move in...	11:
train-100	navi mumbai:4142	3:4142	LiveTV:13,Football:4129	neg	Star Sports 4:13,Manchester United vs Everton:...	
train-1000	new delhi:4131,chennai:2878,navi mumbai:1339	1:658,3:5867,5:413,4:1339,7:71	TalkShow:658,Cricket:7690	neg	SRH vs RCB:701,KKR vs KXIP:1042,MI vs SRH:2288...	
train-40000	gurgaon:6077,chennai:4055	1:1641,2:480,4:1445,7:1663,6:4900	Drama:5503,Cricket:3283,Reality:1345	neg	MI vs KKR:304,Yeh	20:

```
In [122]: ht_dt.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 200000 entries, train-1 to train-99999
Data columns (total 6 columns):
cities     200000 non-null object
dow        200000 non-null object
genres     200000 non-null object
segment    200000 non-null object
titles     200000 non-null object
tod        200000 non-null object
dtypes: object(6)
memory usage: 15.7+ MB
```

Pre-Processing and engineering

- Feature 1 – Genre
  - Separated all the unique genres.
  - Clubbed the redundant genres. Example: Cricket, Boxing - > Sports.
  - Counted occurrence of each genre per segment.
  - New features
    - Genres_count, Travel, Reality, Comedy, ……35.

- Feature 2 – Titles
  - New Features
    - Title_count

- Feature 3 – dow
  - Divide dow into 7 columns. One for each day.
  - New Features
    - Dow_1, dow_2, …….

Pre-Processing and engineering

- Feature 4 – tod
  - Divide time of the day in four quarters.
  - New features
    - tod count
    - First_quarter, Second_quarter……

	title_count	genres_count	cities_count	tod_count	dow_count	Cricket	Tennis	Comedy	Swimming	Athletics	...	dow_2	dow_4	dow_7	dow_3	dow
train-1	60	3	2	14	7	1	0	0	0	0	...	1	1	1	1	
train-10	70	2	3	16	7	1	0	0	0	0	...	1	1	1	1	
train-100	2	2	1	3	1	0	0	0	0	0	...	0	0	0	1	
train-1000	8	2	3	6	5	1	0	0	0	0	...	0	1	1	1	
train-10000	11	3	2	9	5	1	0	0	0	0	...	1	1	1	0	

5 rows × 51 columns

Pre-Processing and engineering

- Features engineering for future modeling
  - Watch time for each genre and weightage
  - Cities
    - Generalizing number of cities by replacing city name with state name in India and country name for else
    - Adding count for top cities for each row
    - We generalized 1358 unique cities into 120 most occurring cities.

Train – Validation – Test Split

+ Data is divided into train and test set with 80% and 20% respectively.
+ Further, train is divided into train and validation set with 70% and 30% respectively.

	Neg	**Pos**	**Total**
Train	103491	8509	112,000
Test	36863	3137	40,000
Validation	44391	3609	48,000

Undersampling – Oversampling using SMOTETomek Technique.

- We have performed Undersampling and Oversampling on train data using SMOTETomek Technique.

	Neg	**Pos**	**Total**
Train	103449	103449	206,898

Model

XGBoost Model:

- Reasons to use:
  1. works well with sparse dataset
  2. works well with large dataset
  3. computationally fast
  4. less prone to overfitting.

XGBoost on dataset without sampling:

```
XGBClassifier trained.

Model Report
--------------------
Accuracy train: 0.9292
AUC Score (Train): 0.537626
train time:
--------------------
Accuracy test: 0.9236
AUC Score (Test): 0.505340
test time:
--------------------
test statistics:
--------------------
             precision    recall  f1-score   support

          0       0.93      1.00      0.96     44391
          1       0.31      0.01      0.03      3609

avg / total       0.88      0.92      0.89     48000

--------------------
[[44287   104]
 [ 3562    47]]
--------------------
--------------------
```

XGBoost on dataset with sampling:

```
XGBClassifier trained.

Model Report
--------------------
Accuracy train: 0.9543
AUC Score (Train): 0.954296
train time:
--------------------
Accuracy test: 0.8795
AUC Score (Test): 0.531123
test time:
--------------------
test statistics:
--------------------
             precision    recall  f1-score   support

          0       0.93      0.94      0.94     44391
          1       0.14      0.12      0.13      3609

avg / total       0.87      0.88      0.87     48000

--------------------
[[41779  2612]
 [ 3172   437]]
--------------------
--------------------
```

Future Work

- Can do more of EDA
- Better Feature Engineering which will contribute to better results.
- Watch_time could be included instead of count.
- Other Machine Learning Models could be tried to know the structure of the data.
- The present XGBoost model can be tuned further.
- Github Profile - https://github.com/swapcy/hackathon_hotstar