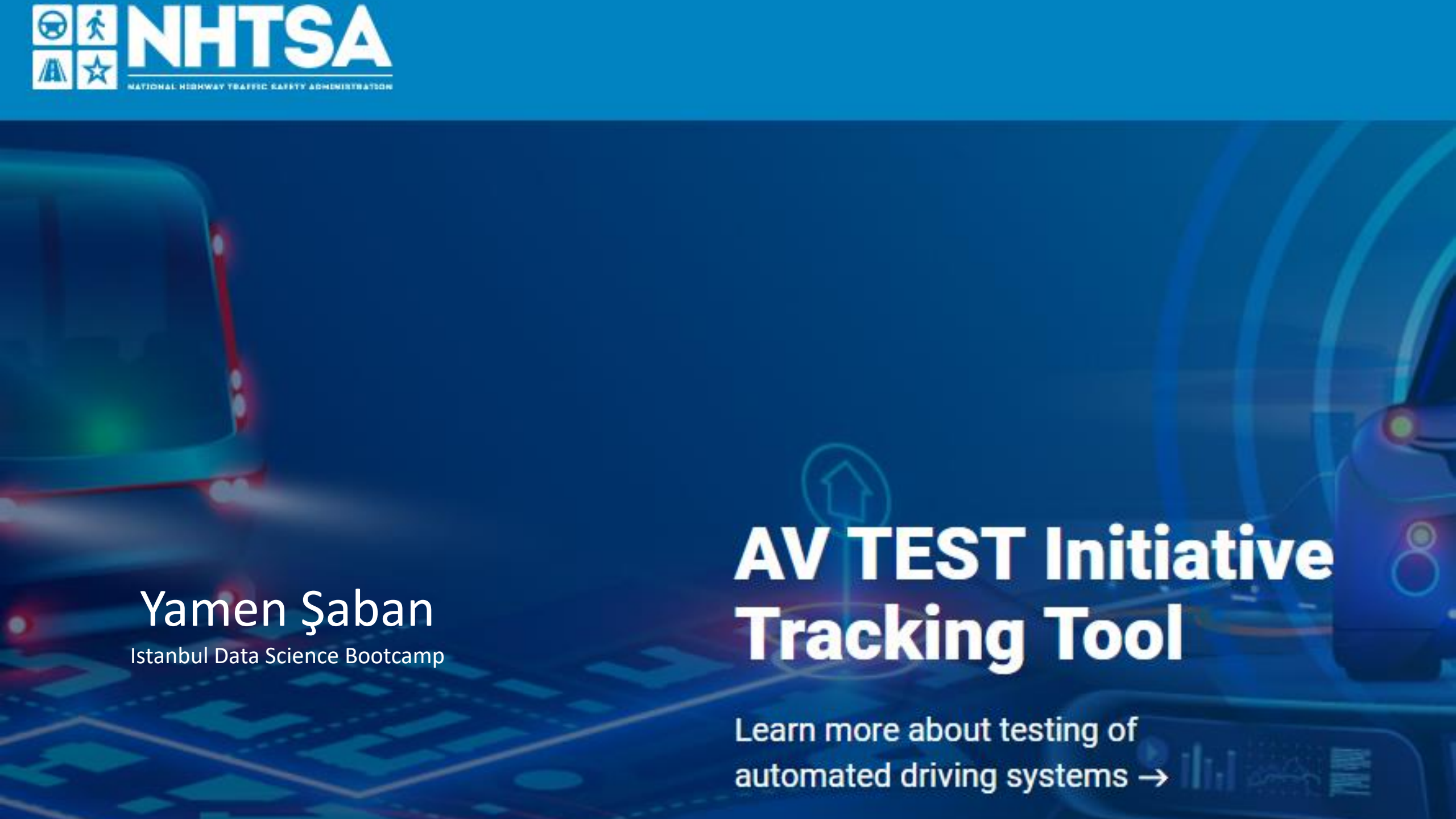


Yamen Şaban
Istanbul Data Science Bootcamp

Project Goal:

The main purpose of this project is to create a model which can help to identify the accidents which had occurred due to a drunk driver.

About the Project:

I. Start with finding a better data source than Kaggle.

II. Getting the EC2 instance ready.

III. Load data into PostgreSQL and create joined views.

IV. Explore the data by applying Exploratory Data Analysis(EDA).

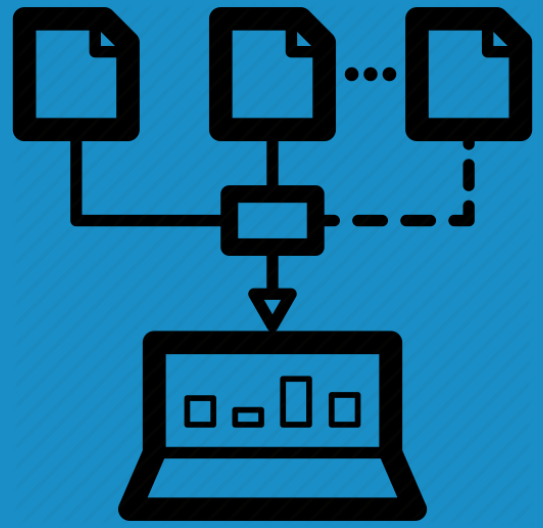
V. Try imbalenced label sampling methods.

VI. Voting approach.

VII. Final Results.

Finding Better Data source:

- If I were only to depend on Kaggle's dataset I would have been left with some old data that goes back to (2015).
- After doing some research I was able to find the real source of data NHTSA and got more recent data (2018).
- And because the data by itself is not enough I had to look to find its documentation on the same site BAAM(done).

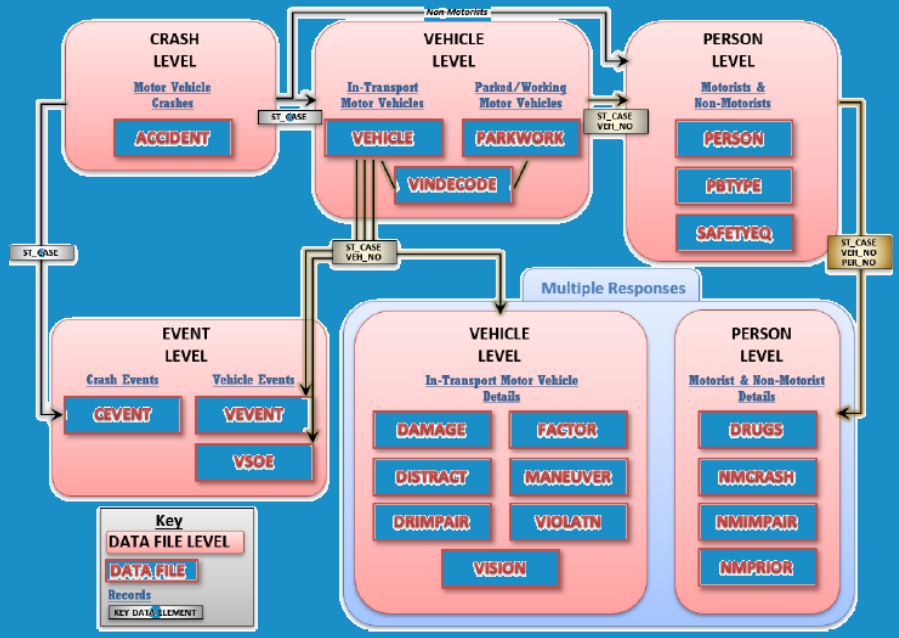


Getting EC2 instance Ready:

- Opening an Amazon Web Services(AWS) account.
- Create an EC2 instance type t2.micro(free).
- I have written a blog about configuring the EC2 to run PostgreSQL, Anaconda, Jupyter Notebook on Medium.
- I will be to storing my data on the EBS volume connected to the instance and run my Python code from the local device on the instance throw tunneling.

Load Data into PostgreSQL:

- After a general overview of the data and depending on the diagram on the right I decided to choose 3 tables of the 27 available once (Accident, Vehicle, Person).
- (Accident → Vehicle → Person) for join.
- Thanks to the data documentation I was able to determine which features to use easily (will be explained in the next slides).



Create Joined Views:

- Because there are (One to Many) relations between the tables I needed to find a way to aggregate features into one line to fit it into the model.
- I had to remove the features not aggregatable.
- No features from the Person table were aggregatable, but I calculated the count of males an females in each car.
- From the Vehicle table I've decided to take the features shown in the picture with suitable aggregation methods.
- Finally create joined views.

```
SELECT sub_vehicle_person.st_case,
    sum(sub_vehicle_person.tow_veh) AS tow_veh,
    max(sub_vehicle_person.vtrafway) AS vtrafway,
    max(sub_vehicle_person.vnum_lan) AS vnum_lan,
    max(sub_vehicle_person.vspd_lim) AS vspd_lim,
    sum(sub_vehicle_person.male_count) AS male_count,
    sum(sub_vehicle_person.female_count) AS female_count
   FROM sub_vehicle_person
  GROUP BY sub_vehicle_person.st_case
```

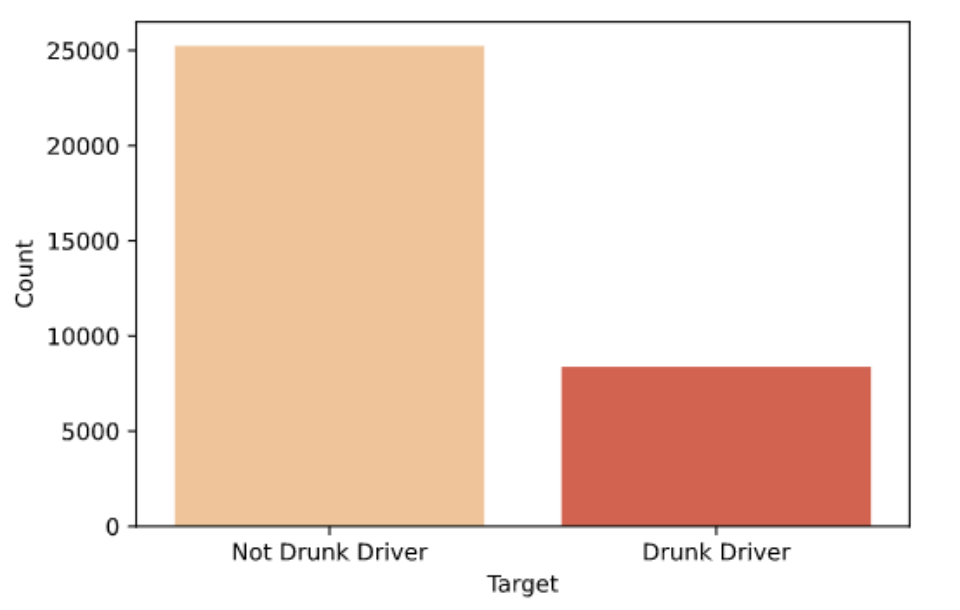
Used Columns:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 33614 entries, 0 to 33613
Data columns (total 31 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   ve_forms       33614 non-null  int32
 1   pvh_invl       33614 non-null  int32
 2   peds           33614 non-null  int32
 3   persons        33614 non-null  int32
 4   county         33614 non-null  int32
 5   day            33614 non-null  int32
 6   month          33614 non-null  int32
 7   day_week       33614 non-null  int32
 8   hour           33614 non-null  int32
 9   nhs            33614 non-null  int32
 10  rur_urb        33614 non-null  int32
 11  func_sys       33614 non-null  int32
 12  rd_owner       33614 non-null  int32
 13  route          33614 non-null  int32
 14  sp_jur         33614 non-null  int32
 15  harm_ev        33614 non-null  int32
 16  man_coll       33614 non-null  int32
 17  typ_int        33614 non-null  int32
 18  wrk_zone       33614 non-null  int32
 19  rel_road       33614 non-null  int32
 20  lgt_cond       33614 non-null  int32
 21  weather        33614 non-null  int32
 22  sch_bus        33614 non-null  int32
 23  cf1            33614 non-null  int32
 24  tow_veh        33614 non-null  int32
 25  vtrafway       33614 non-null  int32
 26  vnum_lan       33614 non-null  int32
 27  vspd_lim       33614 non-null  int32
 28  male_count     33614 non-null  int32
 29  female_count   33614 non-null  int32
 30  drunk_dr       33614 non-null  float64
dtypes: float64(1), int32(30)
memory usage: 4.1 MB
```

Exploratory Data Analysis(EDA):

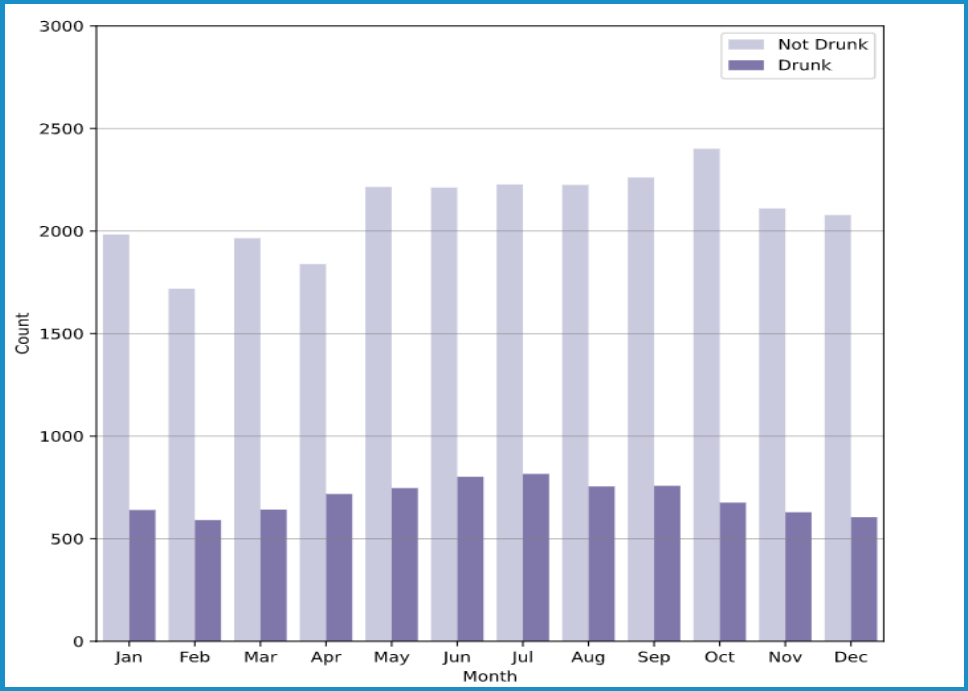
- Exploring the Target Variable:

  After taking a look into the target variable I have found the it contains values(0,1,2,3,4) which represents the number of drunk people involved so in case I wanted to use this column as the target variable I have to map values larger then one to one(Imbalanced HaH!).

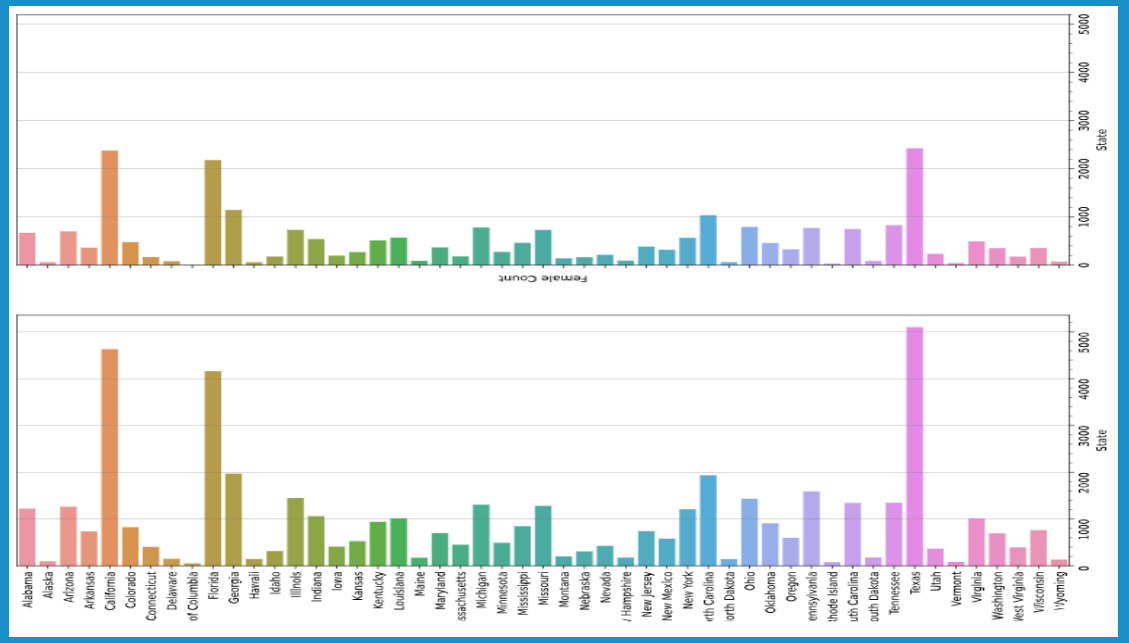


- Exploring the Accidents in Terms of Months:

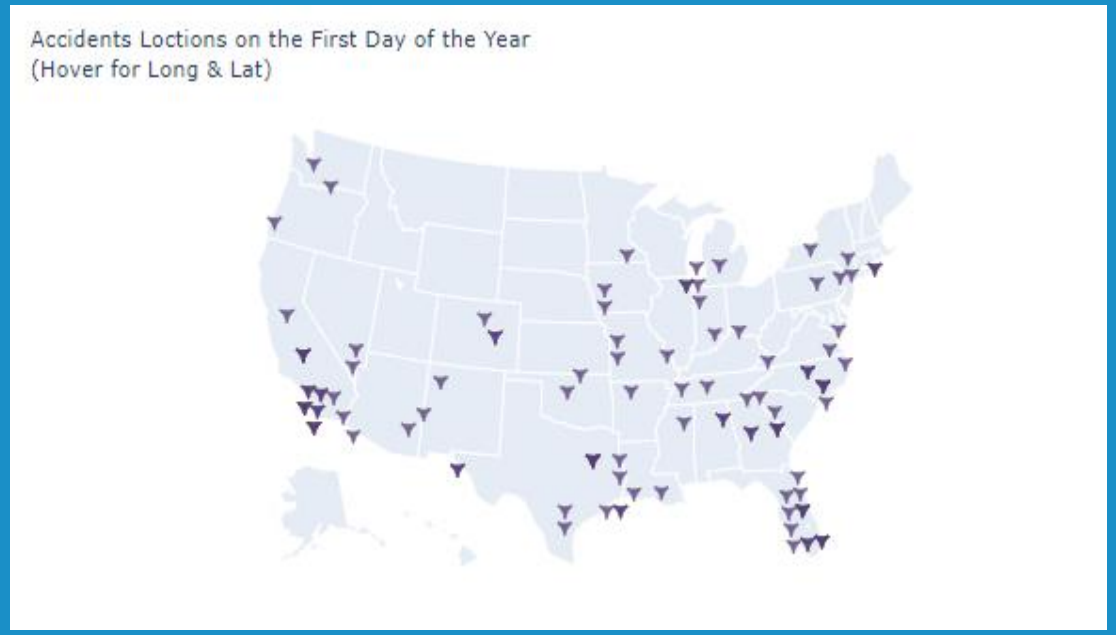
  Number of accidents accrued due to drinking events is rising towards the summer which can be much reasonable(Party All Night).

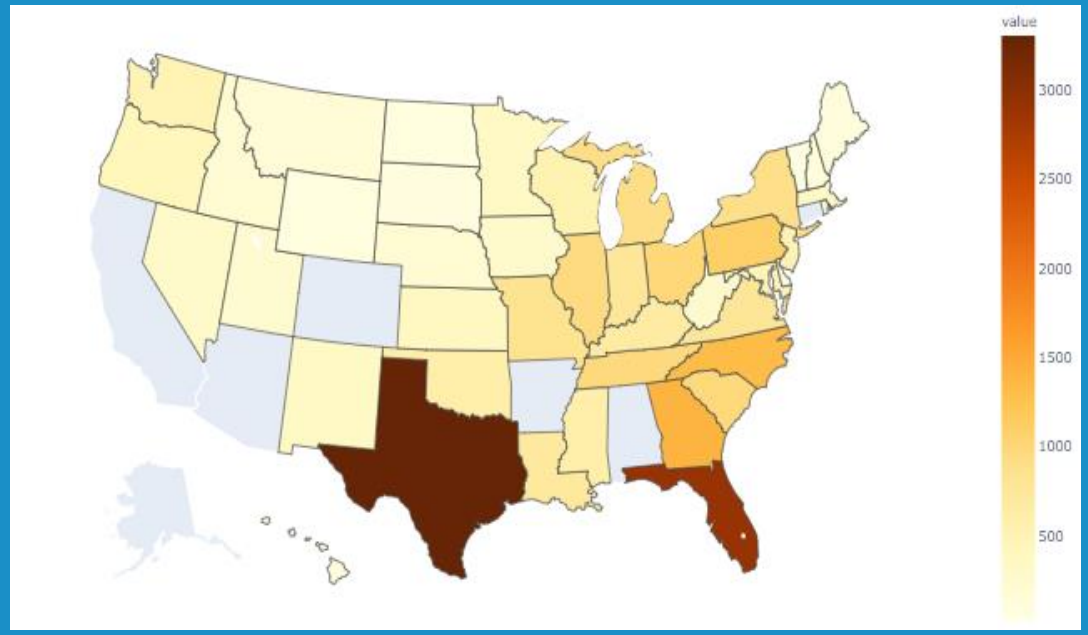


- Exploring the Accidents in Terms of States Male/Female Counts:



- Geo Maps for Better Demonstration:





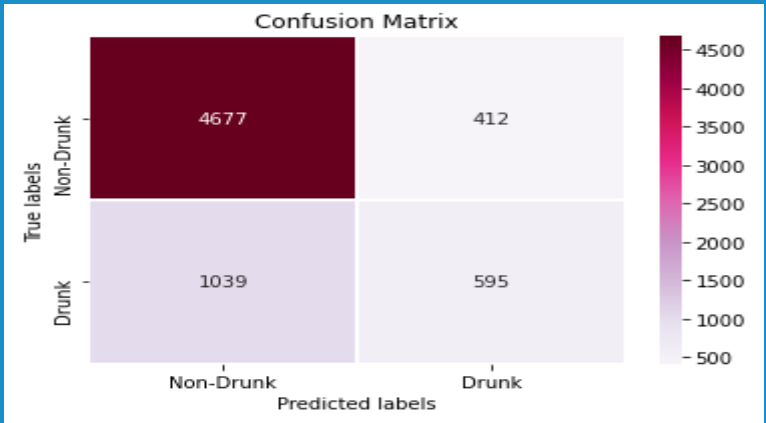
Imbalenced label sampling methods:

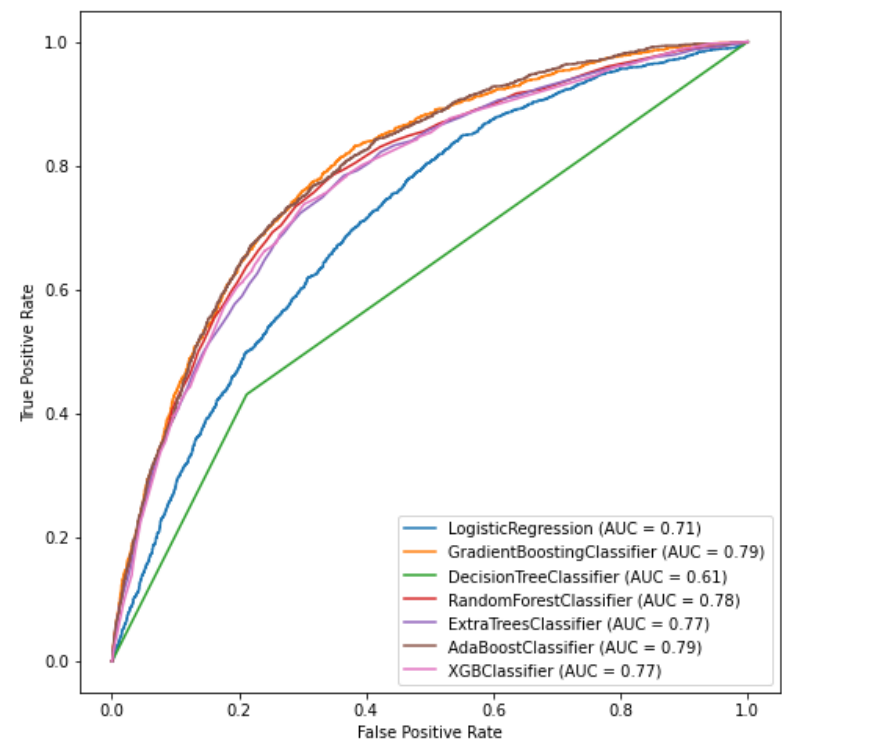
- Training Without Changing:

First I tried to look for the best model which can give me the best results without using any class imbalance methods.
Random Forest Classifier was the winner.

After Cross-Validation I got accuracy: 0.78

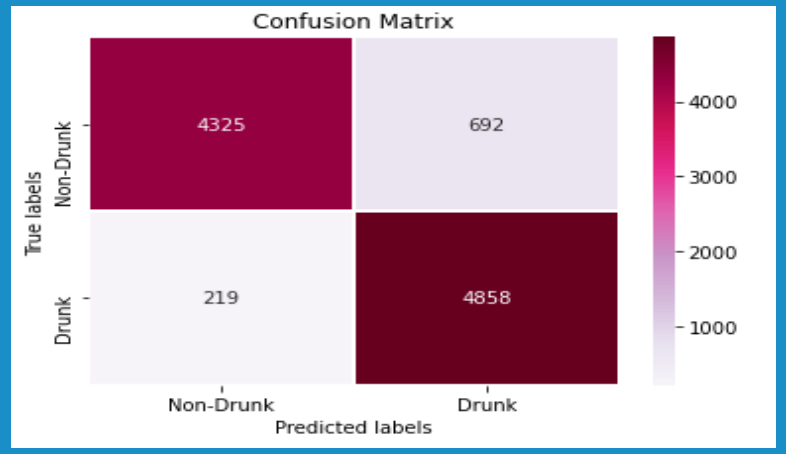
```
----------------------------------------
RandomForestClassifier()
Train accuracy: 0.9999628128370087
Test accuracy: 0.7817938420348058
Precision:0.5815,   Recall:0.3647, F1:0.4483
----------------------------------------
```





- Training With Random Over Sampler:

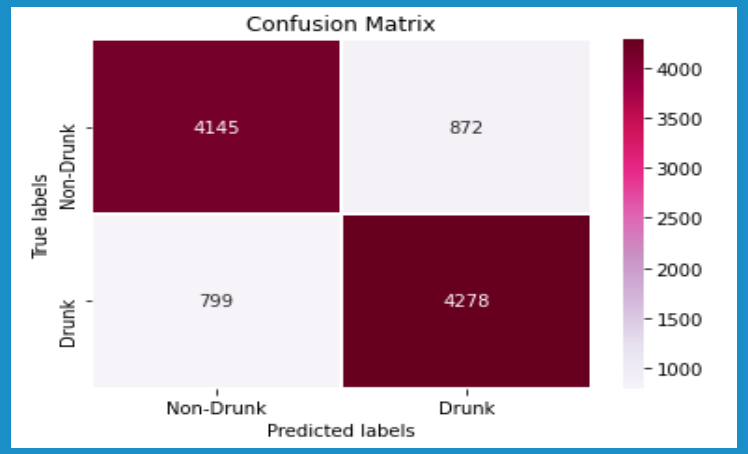
```
----------------------------------------
RandomForestClassifier()
Train accuracy: 1.0
Test accuracy: 0.9080641965524073
Precision:0.8733,   Recall:0.9559, F1:0.9127
----------------------------------------
```



After Cross-Validation I got accuracy: 0.90

- Training With SMOTE:

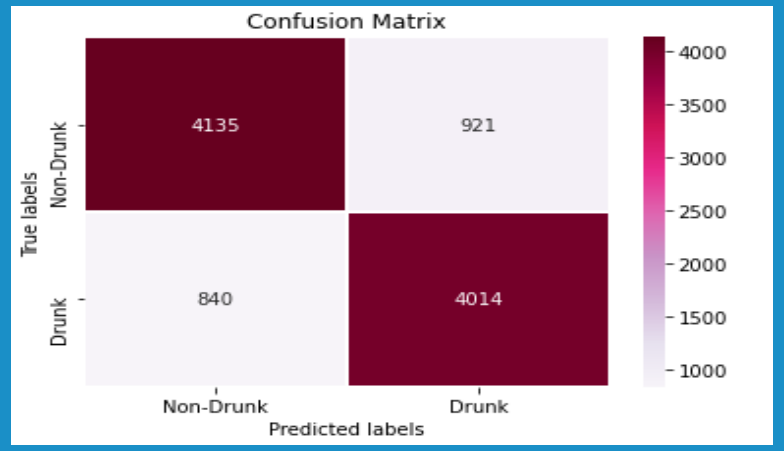
```
----------------------------------------
RandomForestClassifier()
Train accuracy: 1.0
Test accuracy: 0.8360412126015455
Precision:0.8313,   Recall:0.8456, F1:0.8384
----------------------------------------
```



After Cross-Validation I got accuracy: 0.81

- Training With ADASYN:

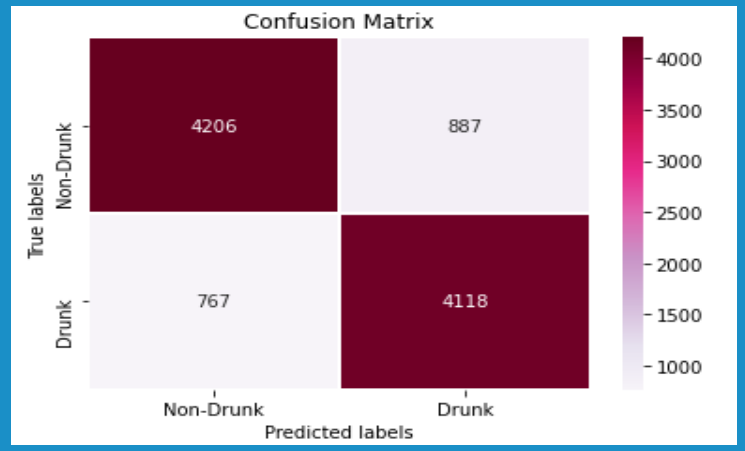
```
----------------------------------------
RandomForestClassifier()
Train accuracy: 1.0
Test accuracy: 0.8260343087790111
Precision:0.8178,   Recall:0.8296, F1:0.8237
----------------------------------------
```



After Cross-Validation I got accuracy: 0.79

- Training With SMOTETOMEK:

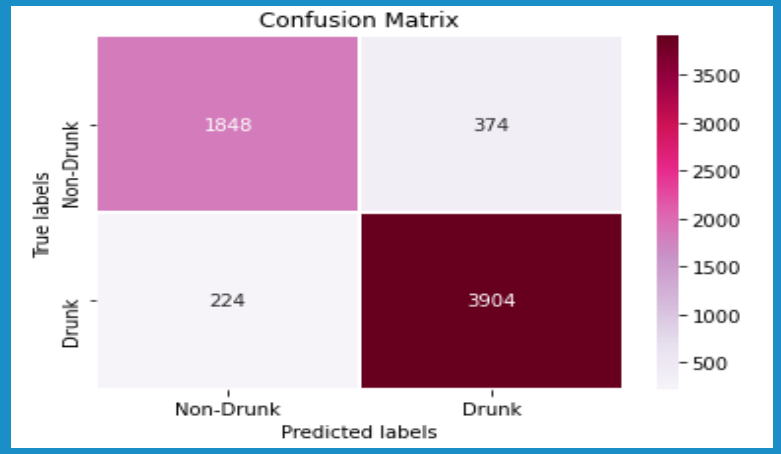
```
----------------------------------------
RandomForestClassifier()
Train accuracy: 1.0
Test accuracy: 0.8300260573261175
Precision:0.8147,   Recall:0.8450, F1:0.8296
----------------------------------------
```



After Cross-Validation I got accuracy: 0.81

- Training With SMOTEENN:

```
----------------------------------------
RandomForestClassifier()
Train accuracy: 1.0
Test accuracy: 0.9069291338582677
Precision:0.9143,   Recall:0.9455, F1:0.9296
----------------------------------------
```



After Cross-Validation I got accuracy: 0.87

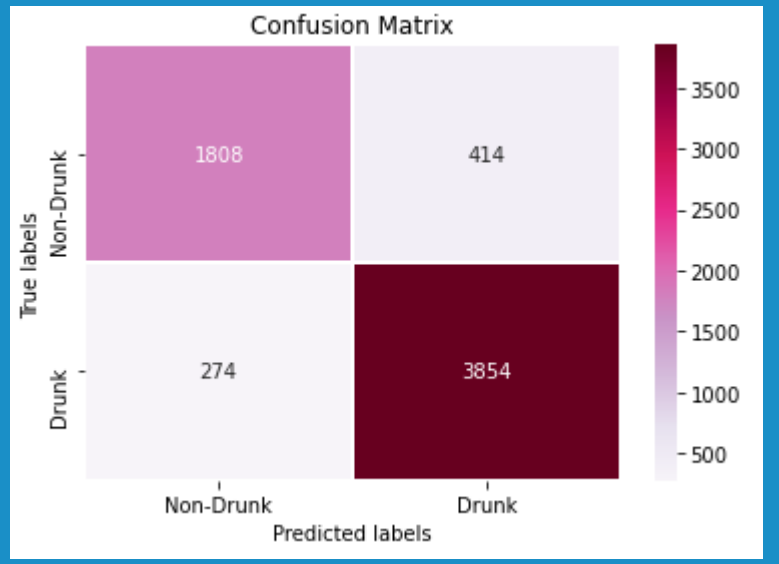
Voting approach:

- Training Using Voting:

  I used 5 of the most succeeded models from before:

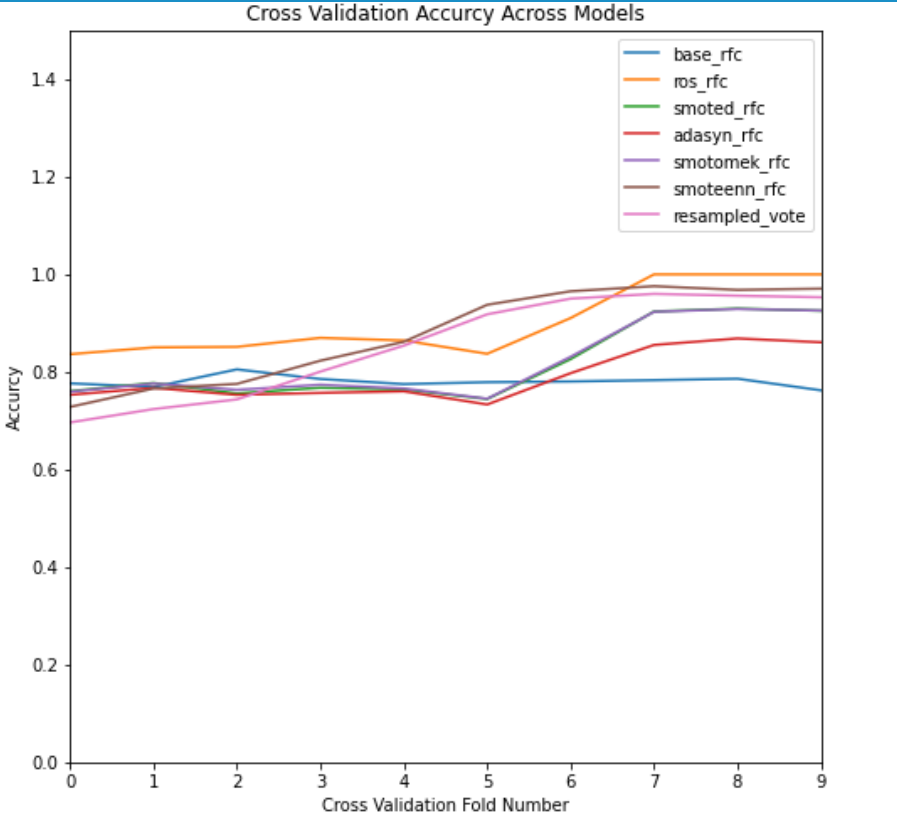
  1. Random Forest Classifier
  2. GradientBoostingClassifier
  3. AdaBoostClassifier
  4. XGBClassifier
  5. ExtraTreesClassifier

  With weights=[5,4,3,2,1]




```
Train accuracy: 0.9256299212598426
Test accuracy: 0.8916535433070866
Precision:0.9030,   Recall:0.9336, F1:0.9181
```

Final Results:



Thank You

LinkedIN: https://www.linkedin.com/in/yamenshaban/

Medium: Medium: https://medium.com/@yamenshabankabakibo

Github: https://github.com/yamenkaba