HACKATHON

AI20BTECH11006 and AI20BTECH11027

Chirag Mehta and Yashas Tadikamalla

Kaggle ID: Account (User ID 8054935)
Username: AI20BTECH11006_AI20BTECH11027
Accuracy: 0.87746

Preprocessing

We dropped the object type columns which had rather irrelevant data, or had a number of distinct values in the order of thousands.
Here is the list of dropped columns

```
'Report Number', 'Local Case Number', 'Road Name', 'Cross-Street Name',
'Off-Road Description', 'Municipality', 'Location', 'Circumstance',
'Vehicle Model', 'Vehicle Make', 'Vehicle ID', 'Road Name', 'Person ID',
'Driverless Vehicle', 'Crash Date/Time', 'Vehicle Year'
```

We combined the entries "Montgomery County Police" and 'MONTGOMERY' in Agency Name, and there were a variety of labels in Drivers Substance Abuse, such as Alcohol Contributed and Alcohol Present, we clubbed them together.
We extracted Day of week, Hour, Month, Year from Crash Date/Time.

Encoding

We tried a lot of encoding techniques, including cat codes, dummies encoding, polynomial encoding, BackwardDifferenceEncoder, BinaryEncoder, CatBoostEncoder, GLMMEncoder, HashingEncoder and many more, the best results were given by polynomial encoding.

Model Used

The basic idea was to stack various ensemble methods. The following base learners were used.

```
    base_learners = [
```

```
                    ('rf_1', RandomForestClassifier(n_estimators=150,
max_features='sqrt')),
                    ('rf_2', LGBMClassifier(boosting_type='dart',
drop_rate=0.6, max_drop=50, n_iter=1000)),
                    ('rf_3', XGBClassifier(use_label_encoder=False,
learning_rate = 0.14, n_estimators=125, max_depth=8))
                   ]
```

The reason behind choosing these is that they perform decent by themselves alone, all achieving 87%+ accuracy on Kaggle.

All parameters are hypertuned to get the maximum accuracy.

Reasons to choose these Classifiers

Random Forest is a standard classifier which gives decent accuracy, but there are models which give better accuracy than random forest which are listed below.

XGBoost: We used it because it is much faster than GradientBoostingClassifier which again gives good results but it is very slow.

LightGBM: light GBM is even faster than XGBoost, the advantage is that there are a lot of parameters to hypertune. It offers dart boosting type which helped reduce chances of overfitting because of drop_rate and max_drop parameters.