IEE 577/ CSE 598: Data Science for System Decision Analytics

Analysis of Machine Learning Models for Road Traffic Accidents

Group 7

Riyank Mukhopadhyay
Aesha Shah
Priyal Padheriya

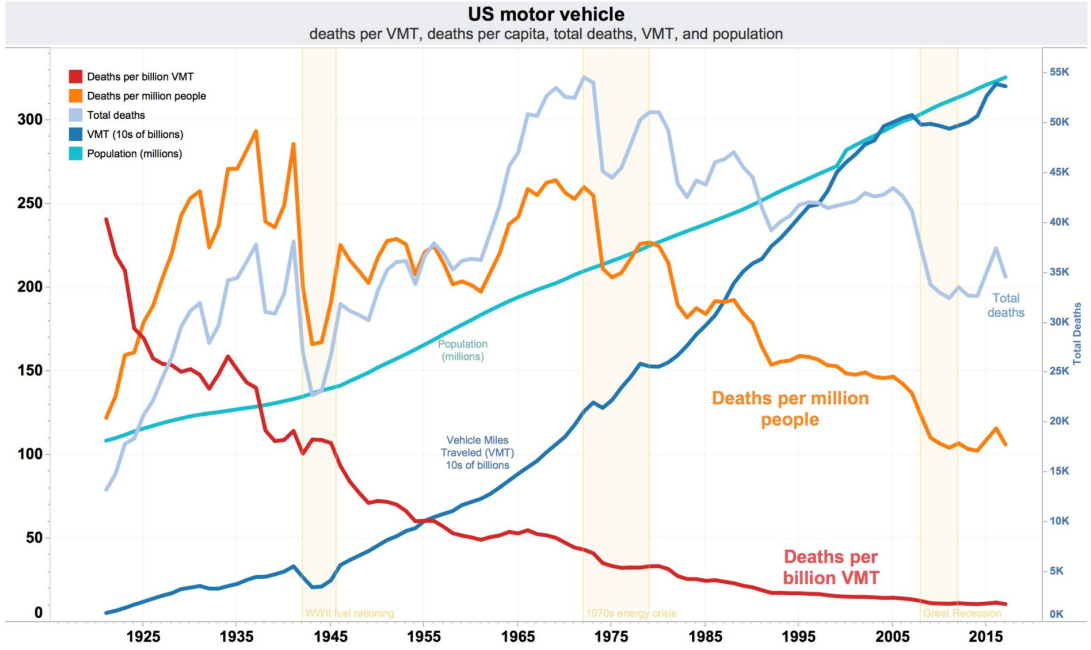
Arizona State University

Arizona State University

Table of Contents

Introduction

- Increasing road safety and reducing traffic accidents have been one of the major factors while developing and maintaining roads.
- Road accidents have an impact on the economy due to loss of life and property damage and also on the population's social and emotional well-being.
- An effective analysis of the severity of the accidents can help in reducing the traffic accidents.
- Also, understanding and analyzing the features that have high impact for Fatal injuries.



Number of Fatalities in the US from 1920 - 2020

Problem Definition

We intend to do an “**Analysis of different machine learning models for Road Traffic Accidents**”.

The problem can be classified into a **Supervised Learning Task** where the Road Traffic Accidents (RTA) dataset will be used for benchmarking the evaluation metrics of various multi-label classification machine learning models.

Dataset

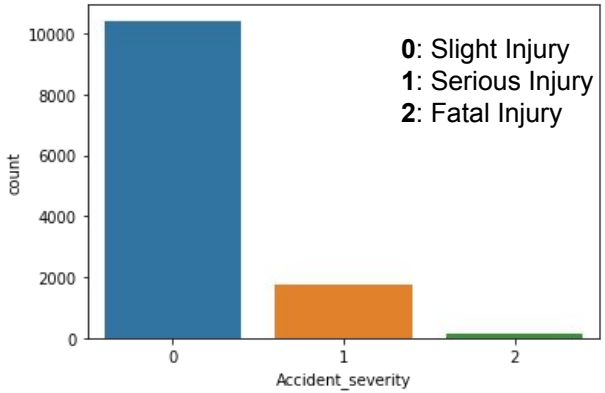
- The data set has been prepared from manual records of road traffic accidents of the year 2017-20.
- All the sensitive information has been excluded during data encoding and finally, it has 32 features and 12316 instances of the accident.
- The **target attribute** is **'Accident_severity'**, which contains **3 categorical** possible values, the values indicate an accident injury severity:
    - Slight Injury
    - Serious Injury
    - Fatal Injury

Time	Day_of_week	Age_band_of_driver	Sex_of_driver
Educational_level	Vehicle_driver_relation	Driving_experience	Type_of_vehicle
Owner_of_vehicle	Service_year_of_vehicle	Defect_of_vehicle	Area_accident_occured
Lanes_or_Medians	Road_allignment	Types_of_Junction	Road_surface_type
Road_surface_conditions	Light_conditions	Weather_conditions	Type_of_collision
Number_of_casualties	Number_of_vehicles_involved	Vehicle_movement	Casualty_class
Sex_of_casualty	Age_band_of_casualty	Casualty_severity	Work_of_casuality
Fitness_of_casuality	Pedestrian_movement	Cause_of_accident	Accident_severity

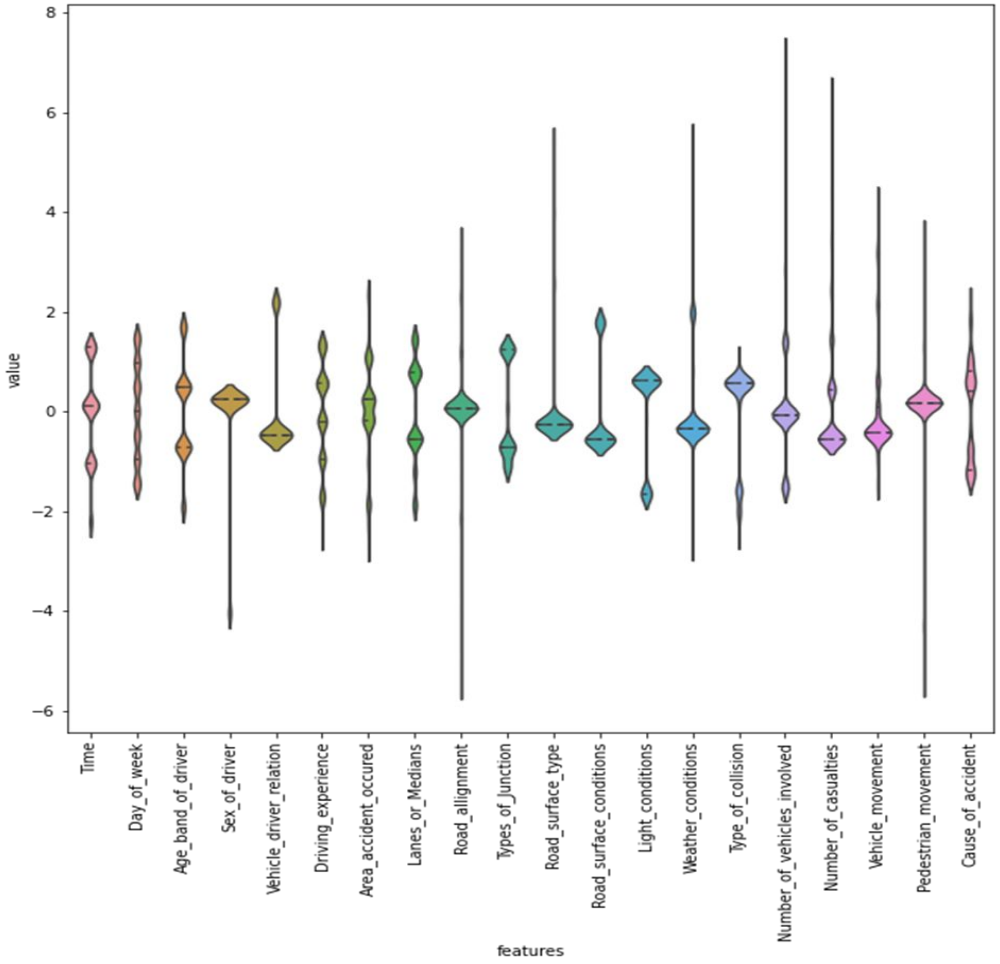
Data Pre-Processing

1. **Selecting useful columns to keep and removing the rest**
    - Predictive - useful
    - Irrelevant / Bad - drop
2. **Handling Null/Missing values**
    - Most frequently used - mode
3. **Converting categorical values**
    - Using map()
4. **Using Label Encoder**
    - Mapping remaining categorical columns to numeric
5. **Train/Test Split**
    - Train - 70%
    - Test - 30%

Data Visualization

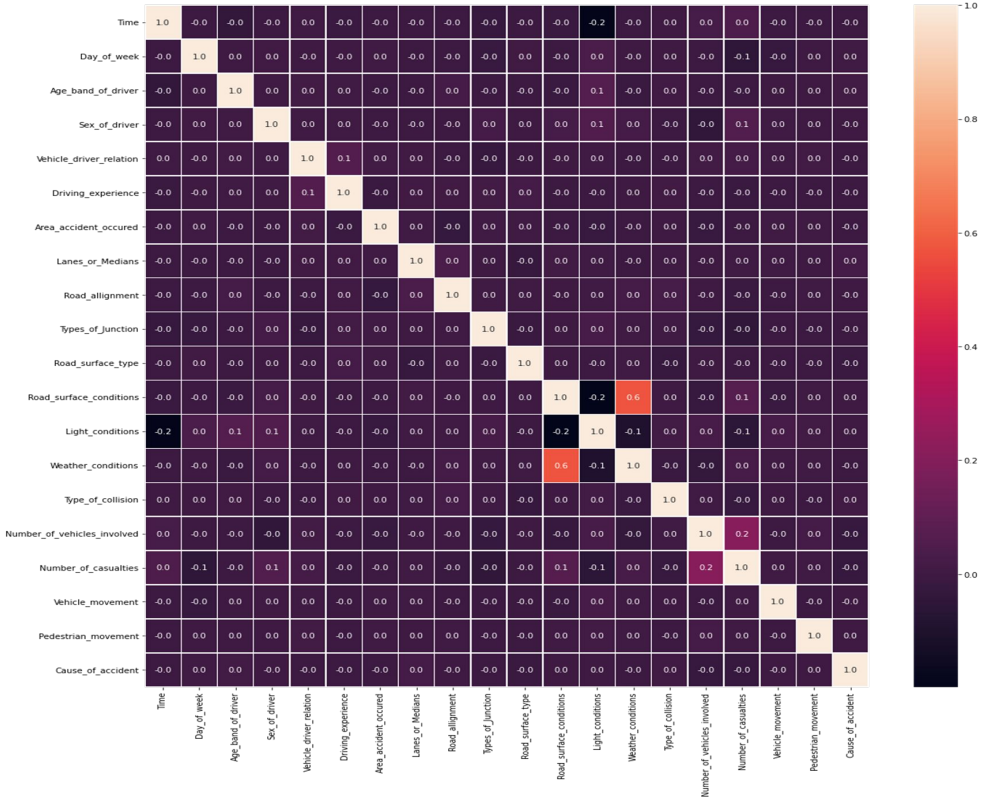


Target Class



Seaborn.violinplot for the 20 features

Data Visualization (contd)



Correlation Map of the 20 features considered

Model Selection

- **Labelled Data**
- **Output Column -** Accident Severity
- **Categories:**
  - Slight Injury
  - Severe Injury
  - Fatal Injury
- **Supervised Learning - Classification Models**
- **Models we used:**
  - Decision Tree Classifier
  - Random Forest Classifier
  - k-Nearest Neighbor Classifier
  - Support Vector Machine Classifier

Methodology

The following Supervised Machine Learning Models were used:

(a) **Decision Trees** - A decision tree contains two different entities - decision nodes and leaves. Decision tree works on top-down approach, i.e. the algorithm starts from the root node and on the basis of the comparison with the dataset, it moves forward to the next node.

(b) **Random Forest -** consists of a finite number of decision trees in various subsets of the dataset from which it takes the prediction and based on a majority vote of predictions it predicts the final outcome. The number of trees present in a random forest is directly proportional to the accuracy of the model and can prevent the issue of overfitting.

(c) **Support Vector Classifier -** has support for both sparse and dense sample vector inputs. The aim of the SVC algorithm is to find a hyperplane in N-dimensional space that clearly classifies the data points. It can be used to find the optimal boundary between the outputs.

(d) **K-Nearest Neighbors -** predict a target variable using one or more independent variables. In this algorithm, a number ‘k’ which is the nearest neighbor to the data point that is to be classified is selected.

Evaluation Metrics

1. **Accuracy** - Ratio of correctly predicted observation to the total observations. Accuracy is a great measure but only when you have symmetric datasets where values of false positive and false negatives are almost same.
    - Accuracy = **TP+TN/TP+FP+FN+TN**
2. **Precision** **-** Ratio of correctly predicted positive observations to the total predicted positive observations.
    - Precision = **TP/TP+FP**
3. **Recall** **-** Ratio of correctly predicted positive observations to the all observations in actual class.
    - Recall = **TP/TP+FN**
4. **F1-score** **-** F1 Score is the weighted average of Precision and Recall, taking into account both false positives and false negatives into account. F1 is usually more useful than accuracy, especially if you have an uneven class distribution.
    - F1 Score = **2*(Recall * Precision) / (Recall + Precision)**

Results - Raw Data

Model Name	Accuracy	Precision	Recall	F1_Score
Decision Tree Classifier	0.753992	0.756981	0.753992	0.755471
K-Nearest Neighbors	0.833829	0.729758	0.833829	0.762248
Random Forest	0.840325	0.818299	0.840325	0.773863
Support Vector Machine Classifier	0.836536	0.699792	0.836536	0.762078

Confusion Matrix for: DecisionTreeClassifier

	precision	recall	f1-score	support
Slight Injury	0.86	0.85	0.85	3091
Serious Injury	0.24	0.25	0.24	552
Fatal injury	0.25	0.27	0.26	52
accuracy			0.75	3695
macro avg	0.45	0.46	0.45	3695
weighted avg	0.76	0.75	0.76	3695

Accuracy: 0.7539918809201623
F1 Score: 0.7554710945303171

Confusion Matrix for: KNeighborsClassifier

	precision	recall	f1-score	support
Slight Injury	0.84	1.00	0.91	3091
Serious Injury	0.20	0.01	0.01	552
Fatal injury	0.00	0.00	0.00	52
accuracy			0.83	3695
macro avg	0.35	0.33	0.31	3695
weighted avg	0.73	0.83	0.76	3695

Accuracy: 0.83382949932341
F1 Score: 0.7622477337463657

Confusion Matrix for: RandomForestClassifier

	precision	recall	f1-score	support
Slight Injury	0.84	1.00	0.91	3091
Serious Injury	0.77	0.04	0.07	552
Fatal injury	0.00	0.00	0.00	52
accuracy			0.84	3695
macro avg	0.54	0.34	0.33	3695
weighted avg	0.82	0.84	0.77	3695

Accuracy: 0.8403247631935047
F1 Score: 0.7738630757300643

Confusion Matrix for: SVC

	precision	recall	f1-score	support
Slight Injury	0.84	1.00	0.91	3091
Serious Injury	0.00	0.00	0.00	552
Fatal injury	0.00	0.00	0.00	52
accuracy			0.84	3695
macro avg	0.28	0.33	0.30	3695
weighted avg	0.70	0.84	0.76	3695

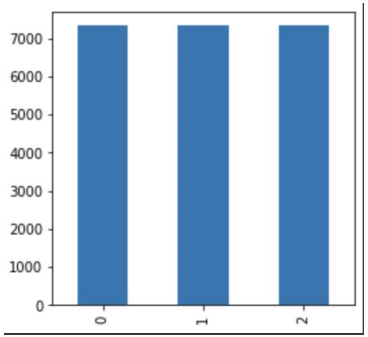
Accuracy: 0.8365358592692829
F1 Score: 0.7620784972005167

Results - Sampling

(a) Under Sampling (US)

Model Name	Accuracy	Precision	Recall	F1_Score
Decision Tree Classifier	0.196482	0.741346	0.196482	0.256775
K-Nearest Neighbors	0.333965	0.722020	0.333965	0.412297
Random Forest	0.122057	0.727126	0.122057	0.145223
Support Vector Machine Classifier	0.162382	0.712062	0.162382	0.230186

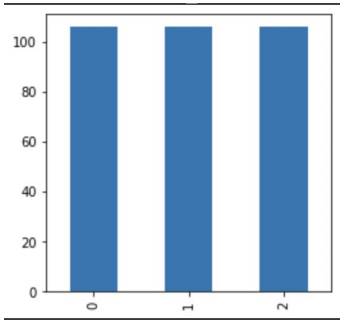
NearMiss()



(b) Over Sampling (US)

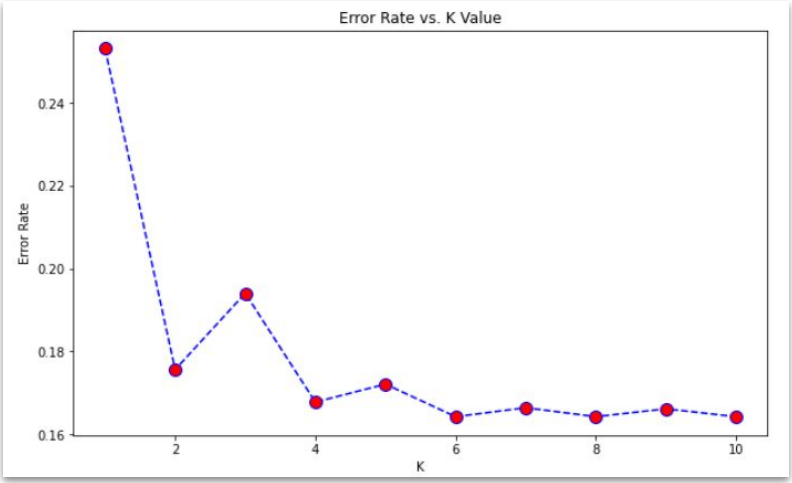
Model Name	Accuracy	Precision	Recall	F1_Score
Decision Tree Classifier	0.674154	0.749384	0.674154	0.705708
K-Nearest Neighbors	0.491204	0.744580	0.491204	0.570039
Random Forest	0.700677	0.755689	0.700677	0.725386
Support Vector Machine Classifier	0.604060	0.738752	0.604060	0.659040

SMOTE()



Fine-tuning: HyperParameter Tuning

(a) KNN



Here, **K = 5** gives the minimum error = **0.1642**.

(b) Random Forest

```
Fitting 3 folds for each of 162 candidates, totalling 486 fits
{'bootstrap': True,
 'max_depth': 15,
 'max_features': 15,
 'min_samples_leaf': 3,
 'min_samples_split': 8,
 'n_estimators': 200}
```

The above set of hyperparameters were obtained after performing **GridSearchCV**

Cross Validation

Cross Validation on best-performing models - **KNN** and **Random Forest**

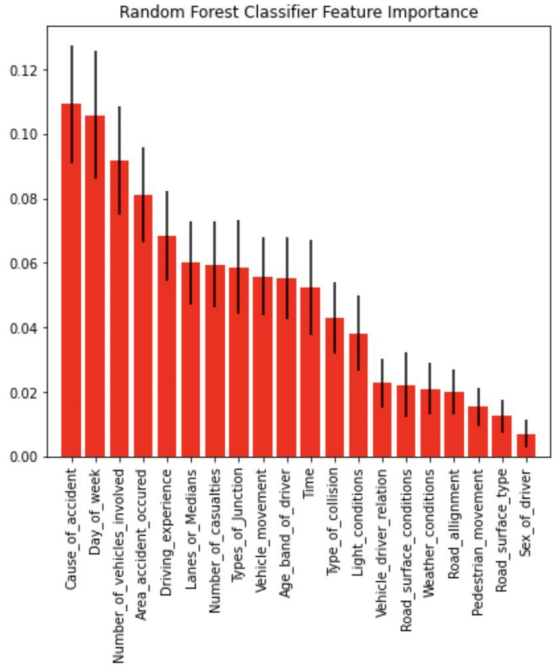
```
def cross_val(model, X_train, y_train, target_names = target_names):
  scoring = ['accuracy', 'precision_weighted', 'recall_weighted', 'f1_weighted']
  kfold = model_selection.KFold(n_splits=10, shuffle=True, random_state=42)
  cv_results = model_selection.cross_validate(model, X_train, y_train, cv=kfold, scoring=scoring)
  clf = model.fit(X_train, y_train)
  y_pred = clf.predict(X_test)

  print(model)
  print('\n', classification_report(y_test, y_pred, target_names=target_names))
  accuracy = accuracy_score(y_test, y_pred)
  f1score = f1_score(y_test, y_pred, average='weighted')
  print("Accuracy:", accuracy)
  print("F1 Score:", f1score)
  print('-----------')
  return clf
```

- KNeighborsClassifier(n_neighbors=5)
- RandomForestClassifier(bootstrap = True, max_depth = 15 ,max_features = 15, min_samples_leaf = 3, min_samples_split = 8, n_estimators=200)]

Feature Importance

We perform Feature Importance on best-performing model - **Random Forest**



Conclusion

Confusion Matrix for Best-performing model -
Decision Tree Classifier after Hyperparameter Tuning and Cross Validation

	precision	recall	f1-score	support
Slight Injury	0.84	1.00	0.91	3091
Serious Injury	0.69	0.07	0.12	552
Fatal injury	1.00	0.02	0.04	52
accuracy			0.84	3695
macro avg	0.85	0.36	0.36	3695
weighted avg	0.82	0.84	0.78	3695

Accuracy: 0.8427604871447902
F1 Score: 0.782937762862057


Any questions?