Name: Lin,junhao

Student ID: 803704353

UPI: jlin897

# Iteration 4 BDAS -Data Mining in traffic accident in UK

## Contents

Github link: https://github.com/jlin897/Infosys722-BDAS

## Step 1 Business Understanding

### 1.1 Identify the objectives of the business/situation

In 2015, United Nations announced sustainable development Goals to transform a bright and sustainable future for all human beings. To address the facing global challenges, there are 17 Goals are proposed, including those related to poverty, inequality, climate change, environmental degradation, peace, and justice. Based on these goals, United Nation called for action by all countries, companies, organizations, and every human being [1].

The sustainable development goals (SDGs) noticed that the trade-off between development and good health and well-being. The SDG 3 committed to ensuring healthy lives and promoting improve the quality of life of all ages is essential to sustainable development. It aims to achieve some targets by 2030, lower the mortality and fertility ratio, ensure universal access quality health care services and affordable medicines, and support the developing countries for risk reduction and management of national.

The SDGs 3 targets 3.6 indicated, halve the number of global deaths and injuries from road traffic accidents by 2030. With rapid development, the roads are shared by cars, buses, trucks, and motorcycles. In recent years, road traffic crashes that responsible for millions of deaths and juries [2]. Road traffic injuries are the eighth causing of death globally for all age groups and are expected to rise 3rd rank by 2020. The number of deaths caused by traffic crashes even greater than HIV and AIDS. According to World Health Organisation data, nearly 1.35 million road traffic deaths globally happened in 2016. For specific, around 3,800 people are killed in traffic crashes and more than 60% of those are walkers, motorcyclists, and cyclists [4].

Traffic accidents not only cause terrible disability and mortality, but also damage and loss of property. It considers being a challenge to improving public health in different countries. Therefore, governments and organizations including the World Health Organization have taken some actions. The WHO encourages governments to prepare a road safe safety strategy that is multisectoral and multidisciplinary. The strategy should consider every possible road user such as the disabled, and it needs to set precise and realistic rules and targets for at least five years. Also, national resources are supposed to well-prepared and support the road safety strategy. For instance, the New Zealand road safety strategy is called Road to Zero 2020 to 2030 [5]. The NZ government set the target of zero which no death or seriously injured while traveling on the land of New Zealand. The government applies the safe system principles and further guiding principles, and it will take actions in five key focus areas. For example, propagate the road safety knowledge such as speed and vehicle safety and improve the road system management. Especially, street lighting systems, road surface conditions and traffic signs are essentially to repair and improve.

- Analysis of situation

In summary, the data related to global car accidents are relatively completed collected by the ministry of transport in different countries. The scientist wishes to use big data to help the driver and Pedestrian understand the details of car accidents and try to prevent some serious accident happened. Based on this researches, some countries proposed some effective road safety strategies. For the effective implementation of this approach, accident prediction and severity prediction are critical. If we can identify the patterns of how these serious accidents happen and the key factors, we might be able to implement well-informed actions and better allocate financial and human resources. In this study, the following goals are proposed:

- To determine the correlation or causality between the Severity of traffic accidents and relevant numbers of factors conditions, weather, speed. What are the key external factors that are most correlated with the severity of traffic accidents? And how effective are they?
- Try to find some pattern of the severity of traffic accidents, such as the time, day of the week.

Tentatively, the study will be judged a success if:

- Successfully determine whether the traffic accident severity is related to road type, time, and the day of the week.
- Achieve a predicting accuracy at least better than random.

## 1.2 Assess the situation

- Requirements

The data used in this study were collected by the ministry of transport in different countries. The ministry of transport will propose a report on traffic accidents quarterly or annually. The open-source data may integrate or modified by the researcher. There are no legal restrictions for my studies. The purpose of the study will provide readers a better perspective and analysis of the traffic accidents, and it will help to achieve the SDG goals of good health and well-being.

- Assumptions

I assumed that there will be not any factors that may affect my study. The data and analysis tools that I used in this study are all open source. Also, the help from the researcher will be gathered on public web platforms. The data that I will be used in my study is gathered from Kaggle, it is a well-known open dataset platform. Therefore, I assume the data source is reliable and completed.

- Constraints

Since the data is gathered from an open-source platform, and the license is the open database. There will be no access to constraint. There are no financial restrictions on this study because and data and tools both are open sources.

- Risk and Contingencies

Data quality is the main concern in this study, Since the records of traffic accidents may vary by different statistics department, the counting variables have different dimensions. This study is trying to involve multiple situation datasets, and it requires good data pre-processing.

Since the limited time and resources, the study may take longer than we expected. Similarly, difficulties can arise depending on the goals of the study. The contingency plan would be spending more time on the research online and seek help for the professional.

It is not surprising if the results were not exactly as we expected. The correlation and causality of traffic accidents are very complex questions. If the result seriously affects the quality and dimensions of the data, the contingency plan will be gathering more data and adjust the model for the study.

## 1.3 Determine data mining objectives

**Data mining goals**

- To determine and rank the key factors which have the biggest impact on the traffic accident severity in the dataset.
- Produce a model that have predicting power for the accident severity of new observations.

**Data mining success criteria**

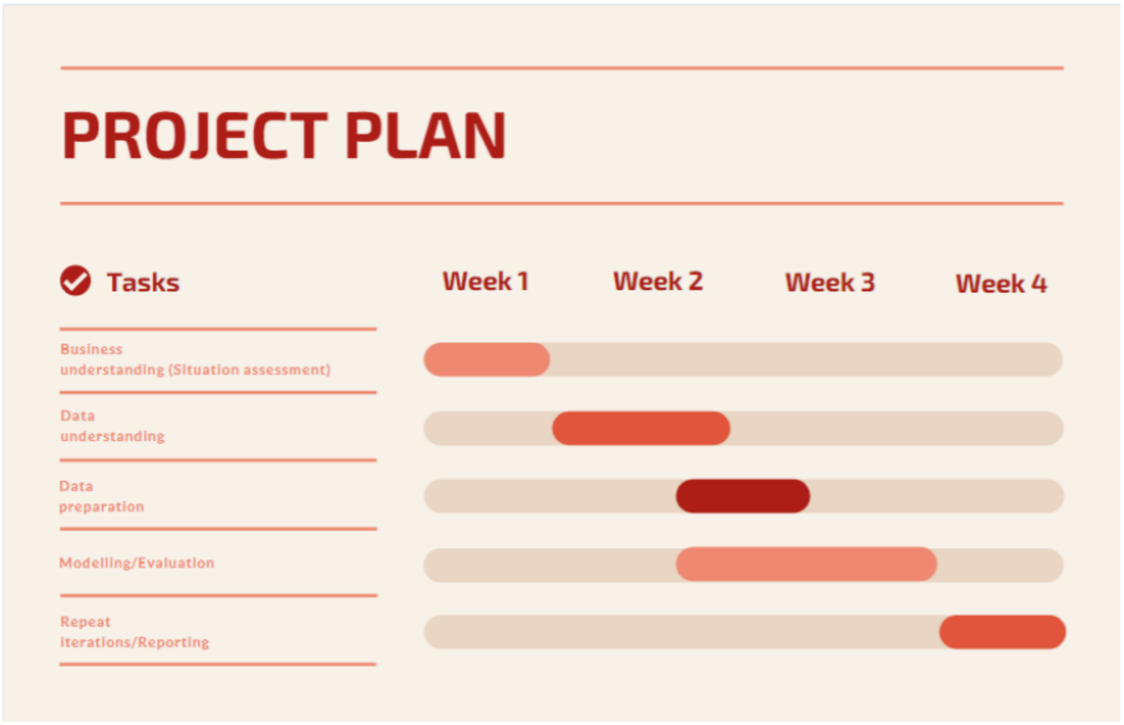
- Successfully determine whether the traffic accident severity is related to road type, time, and the day of the week. A proper regression model for the analysis of the attributes
- Achieve a predicting accuracy at least better than random.

## 1.4 Produce a project plan

| Phase | Duration | Resource |
|---|---|---|
| Business Understanding and Data understanding | Week 1 | All analysts |
| Data preparation | Week 2 | Analysts, Python, Spyder, PySpark |
| Modelling | Week 3 | Analysts, Python, Spyder, PySpark |
| Evaluation | Week 3 | Analysts, Python, Spyder, PySpark |
| Repeat Iterations | Week 4 | Analysts, Python, Spyder, PySpark |
| Reporting | Week 4 | All analysts |

The project plan based on the IBM SPSS Modeler CRISP-DM User Guide [3].

And a Gantt Chart for the project plan:



## Step 2. Data understanding

### 2.1 Collect initial data

The data used in this study were downloaded from Kaggle from a data analyst called Dave Fisher-Hickey(https://www.kaggle.com/daveianhickey/2000-16-traffic-flow-england-scotland-wales). Data were collected and summarised by UK officials. The traffic data were gathered at 1.6 million accidents between 2000 and 2016. The raw data can be download in the UK Department of Transport website(https://roadtraffic.dft.gov.uk/). All the contained accident data were based on the police report, so the data does not include minor accidents.

A glance at the data, the data set was relatively completed, less missing data than we expected, the data were gathered from the police records. However, the missing data is still existing in some attributes. I will treat all the missing data as missing completely at random (MCAR). As I mentioned before, an additional dataset may be involved later, in the case of the unsatisfied model and result due to the quality of data.

### 2.2 Describe the data

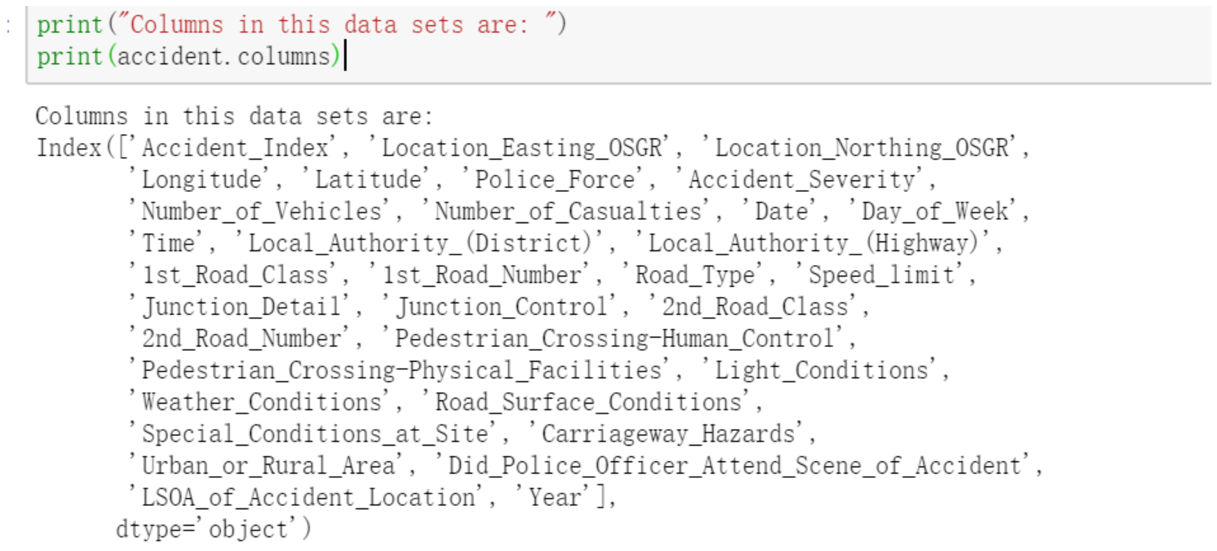
The data consists of all kinds of vehicle collisions from 2000 to 2016. It is a numerical format for all the columns. We can find some support documents that demonstrate every variable involved in the file on www.data.gov.uk website.

- Format of data

The used data format is a CSV file that merges traffic accidents from 2012 to 2014 in the UK.

- Amount of data

The data files contained 464697 observations and 32 variables. The variables will be adjusted during the data pre-processing phase. The dataset contained several different data types, including numeric, categorical, and text. In python, the data types are recognized as integer, floating-point, and object, and the index of the column set to be accident index.



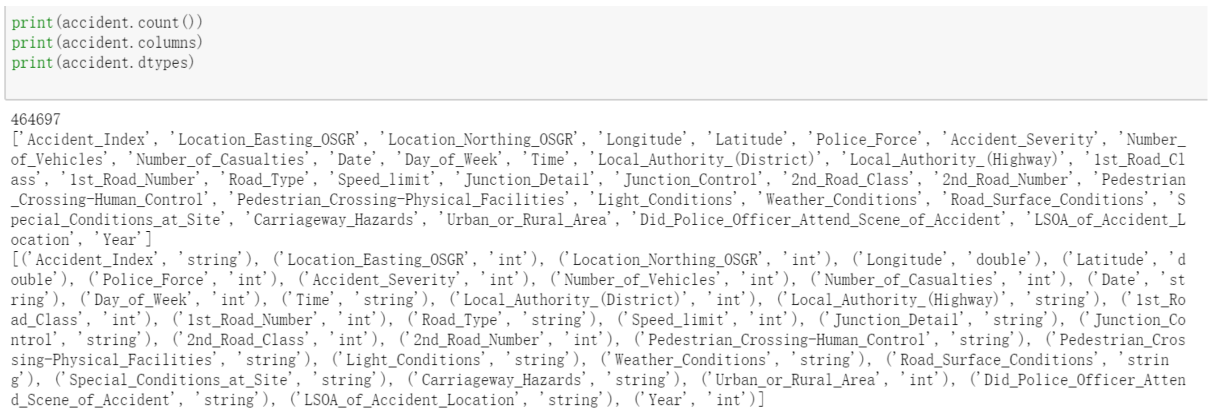
```
print("Columns in this data sets are: ")
print(accident.columns)
```

```
Columns in this data sets are:
Index(['Accident_Index', 'Location_Easting_OSGR', 'Location_Northing_OSGR',
       'Longitude', 'Latitude', 'Police_Force', 'Accident_Severity',
       'Number_of_Vehicles', 'Number_of_Casualties', 'Date', 'Day_of_Week',
       'Time', 'Local_Authority_(District)', 'Local_Authority_(Highway)',
       '1st_Road_Class', '1st_Road_Number', 'Road_Type', 'Speed_limit',
       'Junction_Detail', 'Junction_Control', '2nd_Road_Class',
       '2nd_Road_Number', 'Pedestrian_Crossing-Human_Control',
       'Pedestrian_Crossing-Physical_Facilities', 'Light_Conditions',
       'Weather_Conditions', 'Road_Surface_Conditions',
       'Special_Conditions_at_Site', 'Carriageway_Hazards',
       'Urban_or_Rural_Area', 'Did_Police_Officer_Attend_Scene_of_Accident',
       'LSOA_of_Accident_Location', 'Year'],
      dtype='object')
```

This dataset is relatively large, but it will be adjusted during the data preprocessing step, so the processing time here should not be a major issue.

- Type of data

The dataset used for this study contained several different data types, including numeric, categorical, boolean and text.



```
print(accident.count())
print(accident.columns)
print(accident.dtypes)
```

```
464697
['Accident_Index', 'Location_Easting_OSGR', 'Location_Northing_OSGR', 'Longitude', 'Latitude', 'Police_Force', 'Accident_Severity', 'Number_
of_Vehicles', 'Number_of_Casualties', 'Date', 'Day_of_Week', 'Time', 'Local_Authority_(District)', 'Local_Authority_(Highway)', '1st_Road_Cl
ass', '1st_Road_Number', 'Road_Type', 'Speed_limit', 'Junction_Detail', 'Junction_Control', '2nd_Road_Class', '2nd_Road_Number', 'Pedestrian
_Crossing-Human_Control', 'Pedestrian_Crossing-Physical_Facilities', 'Light_Conditions', 'Weather_Conditions', 'Road_Surface_Conditions', 'S
pecial_Conditions_at_Site', 'Carriageway_Hazards', 'Urban_or_Rural_Area', 'Did_Police_Officer_Attend_Scene_of_Accident', 'LSOA_of_Accident_L
ocation', 'Year']
[('Accident_Index', 'string'), ('Location_Easting_OSGR', 'int'), ('Location_Northing_OSGR', 'int'), ('Longitude', 'double'), ('Latitude', 'd
ouble'), ('Police_Force', 'int'), ('Accident_Severity', 'int'), ('Number_of_Vehicles', 'int'), ('Number_of_Casualties', 'int'), ('Date', 'st
ring'), ('Day_of_Week', 'int'), ('Time', 'string'), ('Local_Authority_(District)', 'int'), ('Local_Authority_(Highway)', 'string'), ('1st_Ro
ad_Class', 'int'), ('1st_Road_Number', 'int'), ('Road_Type', 'string'), ('Speed_limit', 'int'), ('Junction_Detail', 'string'), ('Junction_Co
ntrol', 'string'), ('2nd_Road_Class', 'int'), ('2nd_Road_Number', 'int'), ('Pedestrian_Crossing-Human_Control', 'string'), ('Pedestrian_Cros
sing-Physical_Facilities', 'string'), ('Light_Conditions', 'string'), ('Weather_Conditions', 'string'), ('Road_Surface_Conditions', 'strin
g'), ('Special_Conditions_at_Site', 'string'), ('Carriageway_Hazards', 'string'), ('Urban_or_Rural_Area', 'int'), ('Did_Police_Officer_Atten
d_Scene_of_Accident', 'string'), ('LSOA_of_Accident_Location', 'string'), ('Year', 'int')]
```

- Coding schemes

All categorical variables in the original dataset will be transformed into numeric factors during data processing for modeling purposes in later steps.

## 2.3 Explore the data

Import the original CSV file into Python, there is no cleaning and modifying on the dataset before I import the data.

```
import findspark
findspark.init('/home/ubuntu/spark-2.1.1-bin-hadoop2.7')
import pyspark
from pyspark.sql import SparkSession
spark = SparkSession.builder.appName('BDAS_jlin897').getOrCreate()
```

```
accident = spark.read.csv("accidents_2012_to_2014.csv", header=True, inferSchema= True)
```

- Initial value type check

A quick insight of the data types and values can be done in Python by a function called 'describe' in the 'dataframe' class of the 'Pandas' package. Results comes below with some simple statistics on the values:

```
print(accident[["Accident_Severity"]].describe().show())
```

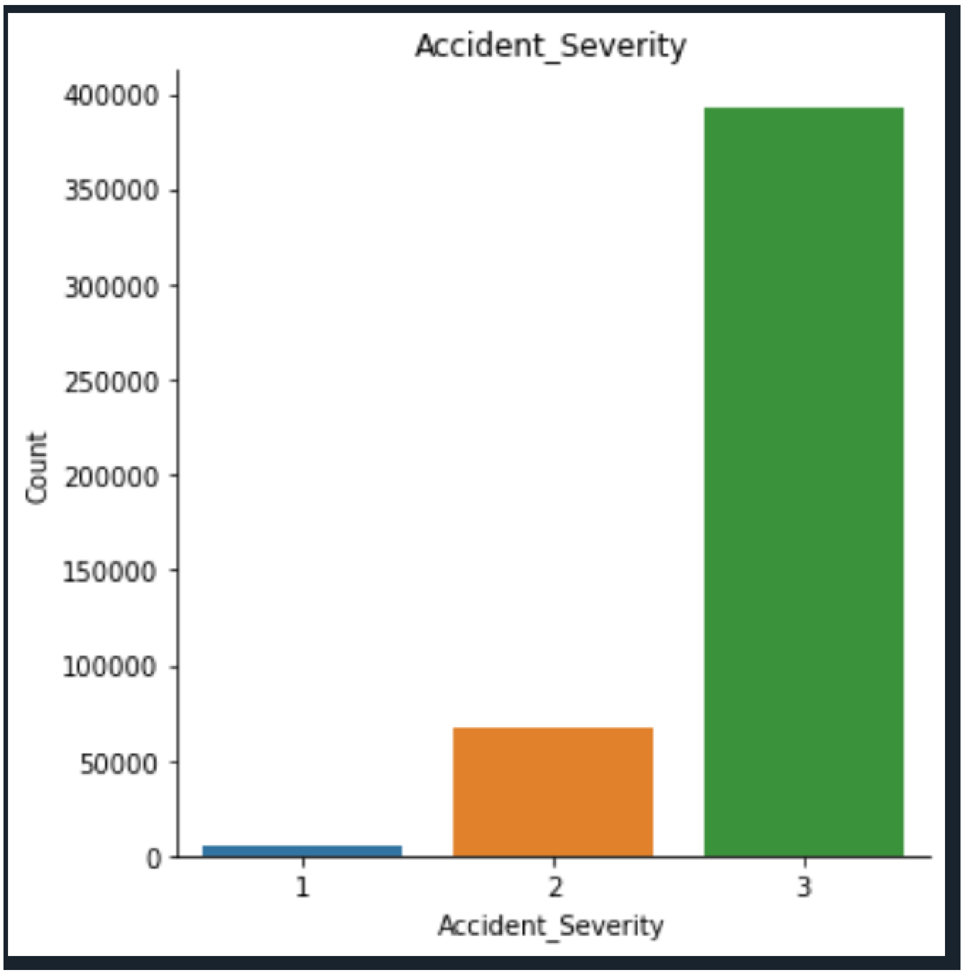
```
+-------+-------------------+
|summary|  Accident_Severity|
+-------+-------------------+
|  count|             464697|
|   mean|  2.833461373755371 8|
| stddev|0.40202902164685816|
|    min|                  1|
|    max|                  3|
+-------+-------------------+
print(accident[["Day_of_Week"]].describe().show())
```

```
+-------+------------------+
|summary|       Day_of_Week|
+-------+------------------+
|  count|            464697|
|   mean| 4.108739673378567|
| stddev|1.9164285644856303|
|    min|                 1|
|    max|                 7|
+-------+------------------+
```

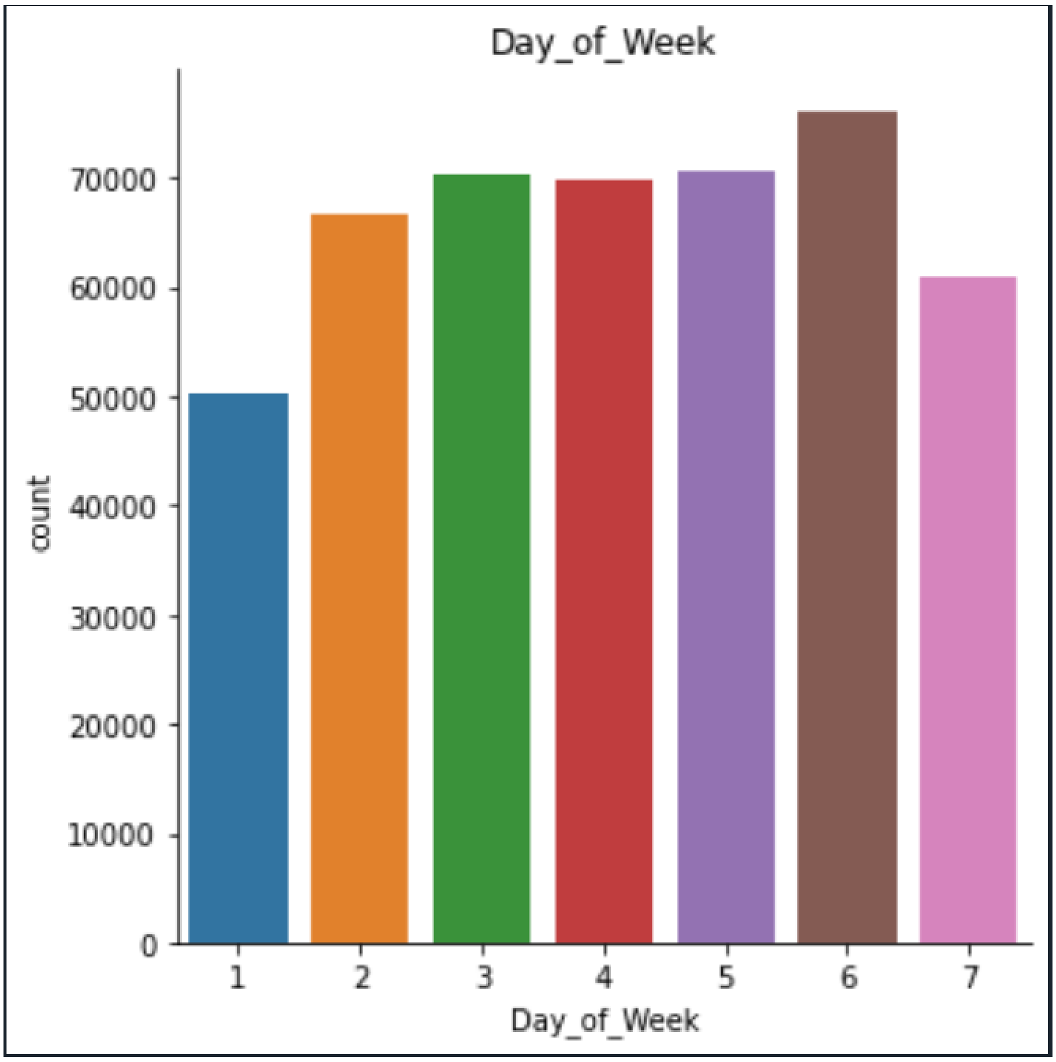
Most of the variables are recognize to be object variables in Python. Some of the columns are string, therefore the statistics of those columns show NaN.

- Initial data visualisation

  By plotting the histogram and count the frequency of the variables. We can have look at some of the distribution of the observations in this dataset.



There are 3 levels of severities, from the highest severity 1 to lowest severity 3. The distribution histogram of accident severity shows that most of the traffic accidents were low severity at the level of 3, there are around 390,000 cases classified into low severity. About 80,000 cases were classified into medium severity, and very few accidents were high severity.

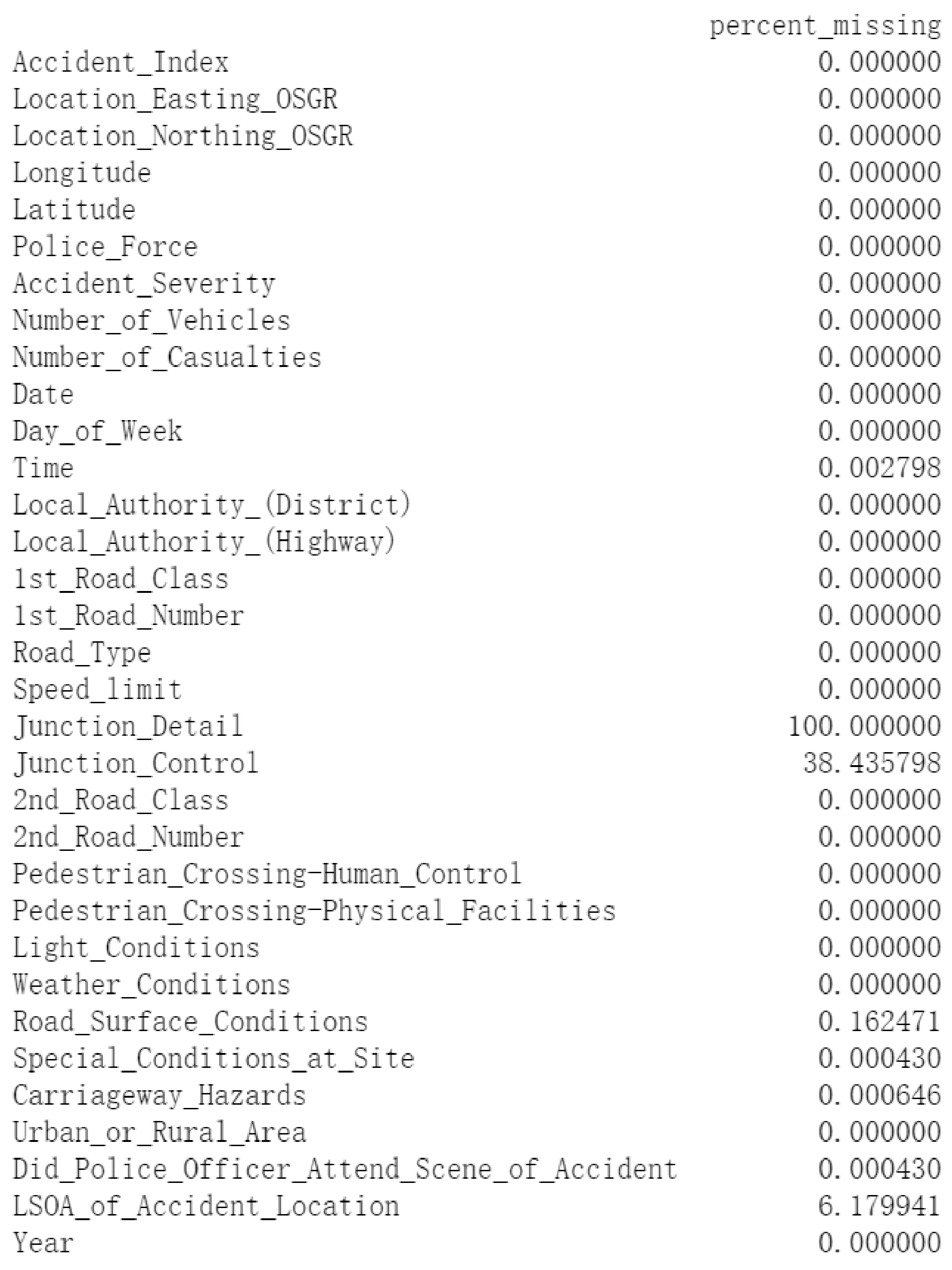


From the distribution histogram of the day of the week with accident severity. In this dataset, the Day of the week starts on Sunday, so 1 to 7 refer to Sunday to Saturday respectively. We can see that the greatest number of traffic accidents that happened was Friday, and the lowest number of accidents was on Saturday and Sunday in this dataset. And the rest of the day of the week has a relatively close count of accidents.

2.4 Verify the data quality

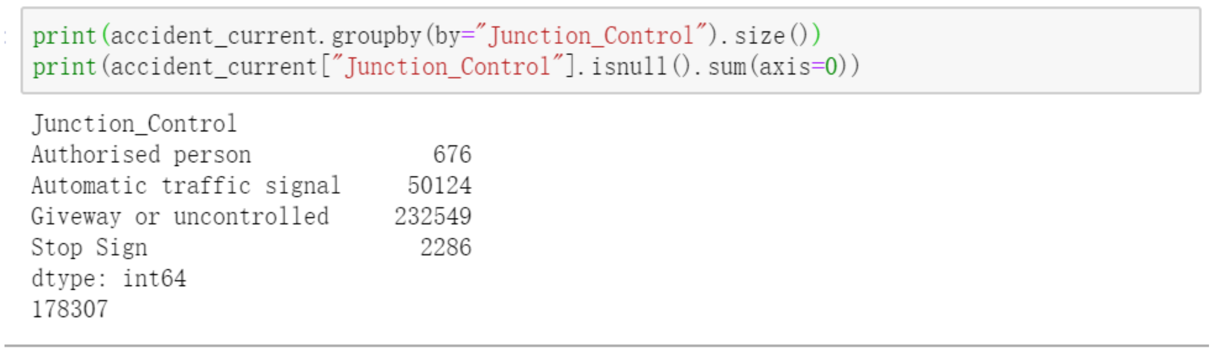
In data analysis, the quality of the data set is particularly important. Because the relationship between the quality of the data set and the analysis results is positively correlated. In other words, if the quality of the data set is not high, the analysis that can be done is very limited.

- Missing data



```
                                           percent_missing
Accident_Index                                    0.000000
Location_Easting_OSGR                             0.000000
Location_Northing_OSGR                            0.000000
Longitude                                         0.000000
Latitude                                          0.000000
Police_Force                                      0.000000
Accident_Severity                                 0.000000
Number_of_Vehicles                                0.000000
Number_of_Casualties                              0.000000
Date                                              0.000000
Day_of_Week                                       0.000000
Time                                              0.002798
Local_Authority_(District)                        0.000000
Local_Authority_(Highway)                         0.000000
1st_Road_Class                                    0.000000
1st_Road_Number                                   0.000000
Road_Type                                         0.000000
Speed_limit                                       0.000000
Junction_Detail                                 100.000000
Junction_Control                                 38.435798
2nd_Road_Class                                    0.000000
2nd_Road_Number                                   0.000000
Pedestrian_Crossing-Human_Control                 0.000000
Pedestrian_Crossing-Physical_Facilities           0.000000
Light_Conditions                                  0.000000
Weather_Conditions                                0.000000
Road_Surface_Conditions                           0.162471
Special_Conditions_at_Site                        0.000430
Carriageway_Hazards                               0.000646
Urban_or_Rural_Area                               0.000000
Did_Police_Officer_Attend_Scene_of_Accident       0.000430
LSOA_of_Accident_Location                         6.179941
Year                                              0.000000
```

Most of the variables are completed in this dataset. The quality of data is not perfect, but it can be adjusted by the data pre-processing phase, and even use an additional dataset. The missing data include NULL values, Empty string, White space, and Blank Value.



```
print(accident_current.groupby(by="Junction_Control").size())
print(accident_current["Junction_Control"].isnull().sum(axis=0))
```

```
Junction_Control
Authorised person             676
Automatic traffic signal    50124
Giveway or uncontrolled    232549
Stop Sign                    2286
dtype: int64
178307
```

Most of the data fields are 100% complete, except for the control of Junction and detail of Junction. It could explain by hard to defined the junction control of the scene of the accident, or it just an undefined statement. There are very few missing data for Time, condition of the road surface, and hazards of the carriageway. The empty string could be undefined from the original police report, Therefore the columns leave it empty.

- Data errors

It must exist some of the measurement errors by bad measurement or definition. in this dataset, but most of these are not associated variables, therefore measurement error should not be concerned in this study. Times involved some null value, it may be caused by inconsistent value input, which Python misclassify to the null value.

- Measurement errors

Measurement errors include data entered correctly but based on the wrong measurement scheme. There are no related variables in this dataset, so measurement errors should not be overly concerned.

- Bad Metadata

The field name and definition are unclear in this dataset, the data dictionary needs to be check from the source document. Data field such as Local authority for distinct and highway are confused, there is no details and explanation for these field from the data source.

## Step 3 Data Preparation

### 3.1 Select data

- Selecting items

The goal of our study is to find the relationship between traffic accident casualties and other factors such as time and road conditions. Therefore, I need to identify the valid observation from my original data.

Among the patient data I have, there is a filed named "Weather condition" and “road surface condition”. Most of the observations have a valid value, but there are still thousands of records that were undefined, unknown, or empty. So, I need to subset my patient data as shown below:

```
Road_Surface_Conditions
Dry                          319370
Flood (Over 3cm of water)       863
Frost/Ice                      8140
Snow                           2824
Wet/Damp                     132745
dtype: int64
```

```
accident_current = accident[accident["Road_Surface_Conditions"].notnull() &
                            (accident["Weather_Conditions"].notnull())]
print("The dimension of the traffic accident 2012 to 2014 dataset is "+ str(accident_current.shape[0]) +
      " rows and " + str(accident_current.shape[1]) + " columns.")
```

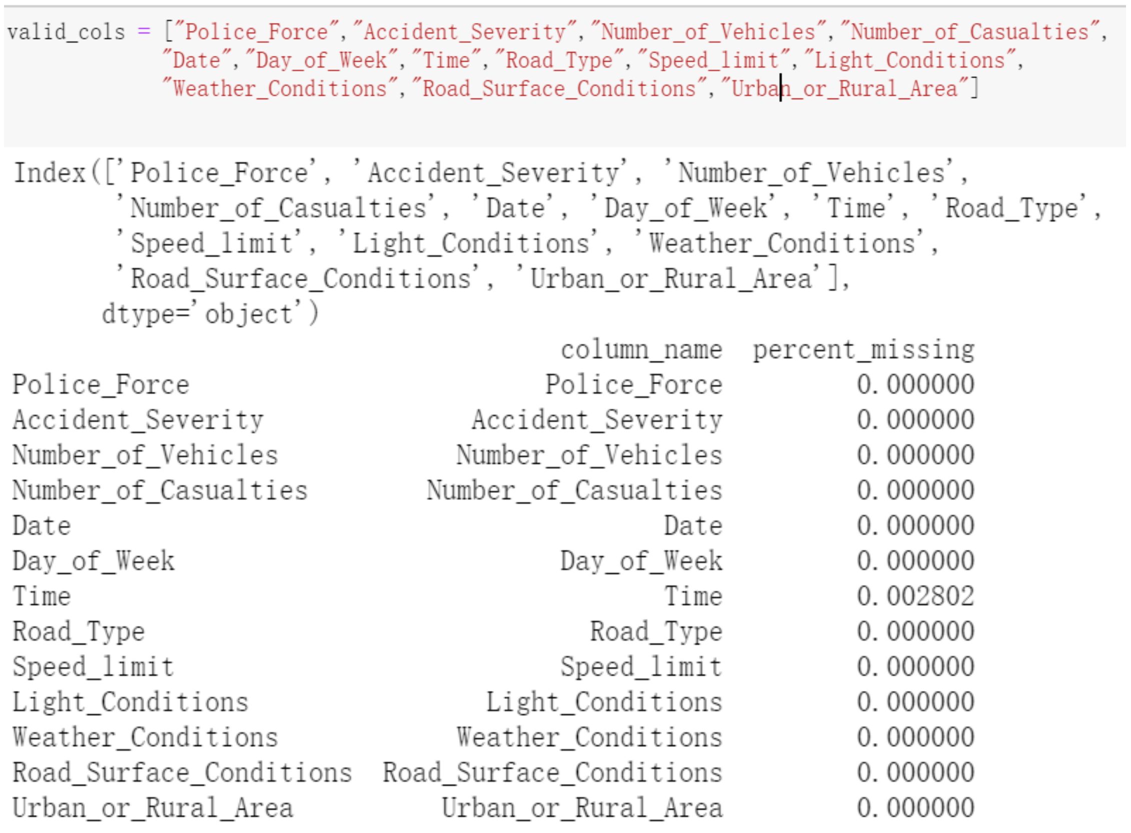
```
The dimension of the traffic accident 2012 to 2014 dataset is 463942 rows and 33 columns.
```

And the result of the missing percentage and checking for null value after select the items

```
                                              percent_missing
Accident_Index                                       0.000000
Location_Easting_OSGR                                0.000000
Location_Northing_OSGR                               0.000000
Longitude                                            0.000000
Latitude                                             0.000000
Police_Force                                         0.000000
Accident_Severity                                    0.000000
Number_of_Vehicles                                   0.000000
Number_of_Casualties                                 0.000000
Date                                                 0.000000
Day_of_Week                                          0.000000
Time                                                 0.002802
Local_Authority_(District)                           0.000000
Local_Authority_(Highway)                            0.000000
1st_Road_Class                                       0.000000
1st_Road_Number                                      0.000000
Road_Type                                            0.000000
Speed_limit                                          0.000000
Junction_Detail                                    100.000000
Junction_Control                                    38.433037
2nd_Road_Class                                       0.000000
2nd_Road_Number                                      0.000000
Pedestrian_Crossing-Human_Control                    0.000000
Pedestrian_Crossing-Physical_Facilities              0.000000
Light_Conditions                                     0.000000
Weather_Conditions                                   0.000000
Road_Surface_Conditions                              0.000000
Special_Conditions_at_Site                           0.000216
Carriageway_Hazards                                  0.000431
Urban_or_Rural_Area                                  0.000000
Did_Police_Officer_Attend_Scene_of_Accident          0.000216
LSOA_of_Accident_Location                            6.187196
Year                                                 0.000000
```

- Select attributes(filed)

For the goal of my study, there are some attributes not need to include in the analysis such as index, longitude, and latitude. However, few possible attributes could potentially have a huge impact on accident severity such as road condition and weather conditions. We can apply the Filter node to discard the attributes. And there are 13 attributes left after filtering.



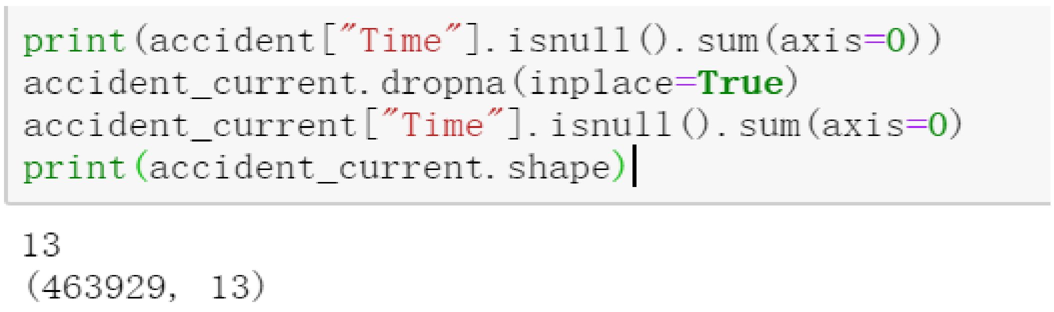
```
valid_cols = ["Police_Force","Accident_Severity","Number_of_Vehicles","Number_of_Casualties",
              "Date","Day_of_Week","Time","Road_Type","Speed_limit","Light_Conditions",
              "Weather_Conditions","Road_Surface_Conditions","Urban_or_Rural_Area"]
```

```
Index(['Police_Force', 'Accident_Severity', 'Number_of_Vehicles',
       'Number_of_Casualties', 'Date', 'Day_of_Week', 'Time', 'Road_Type',
       'Speed_limit', 'Light_Conditions', 'Weather_Conditions',
       'Road_Surface_Conditions', 'Urban_or_Rural_Area'],
      dtype='object')
                                    column_name  percent_missing
Police_Force                       Police_Force         0.000000
Accident_Severity             Accident_Severity         0.000000
Number_of_Vehicles           Number_of_Vehicles         0.000000
Number_of_Casualties       Number_of_Casualties         0.000000
Date                                       Date         0.000000
Day_of_Week                         Day_of_Week         0.000000
Time                                       Time         0.002802
Road_Type                             Road_Type         0.000000
Speed_limit                         Speed_limit         0.000000
Light_Conditions               Light_Conditions         0.000000
Weather_Conditions           Weather_Conditions         0.000000
Road_Surface_Conditions Road_Surface_Conditions         0.000000
Urban_or_Rural_Area         Urban_or_Rural_Area         0.000000
```

3.2 Clean the data

- Missing data

This data has a high degree of completion. All fields except time have 100% completion. We simply drop out the missing value in Time.

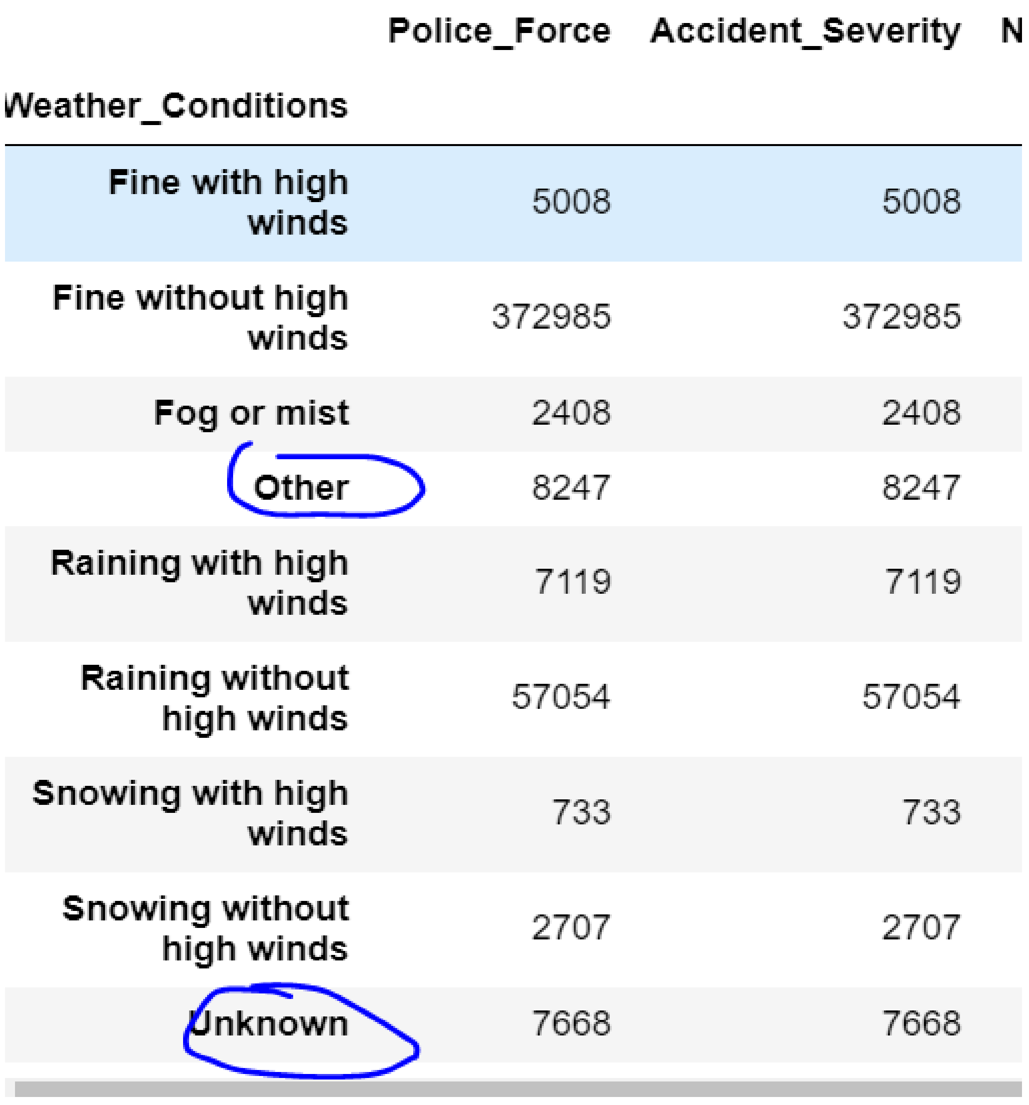


```
print(accident["Time"].isnull().sum(axis=0))
accident_current.dropna(inplace=True)
accident_current["Time"].isnull().sum(axis=0)
print(accident_current.shape)
```

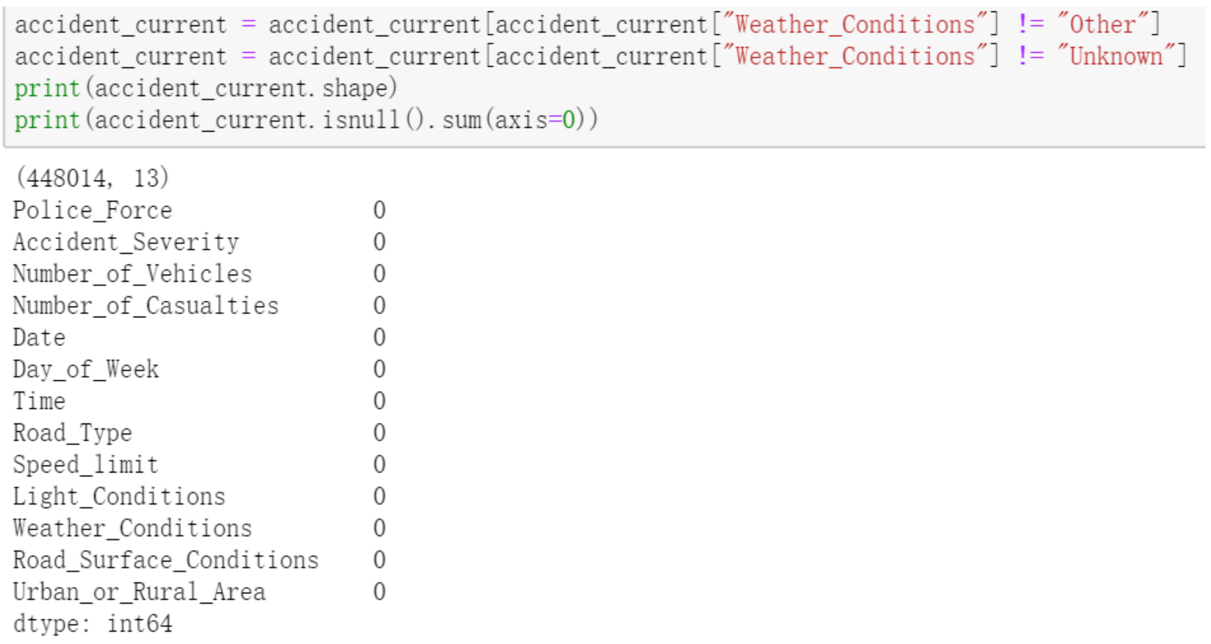
```
13
(463929, 13)
```

There are 13 missing observations. After drop out the missing value in Time, the observations were reduced from 463942 to 463929.

- Data error
  Some attributes have an empty string and undefined value. Since its low proportion of the error data, we can simply discard such data by removing the rows with undefined value such as unknown or other in weather conditions.



| | Police_Force | Accident_Severity | N |
|---|---|---|---|
| Weather_Conditions | | | |
| Fine with high winds | 5008 | 5008 | |
| Fine without high winds | 372985 | 372985 | |
| Fog or mist | 2408 | 2408 | |
| Other | 8247 | 8247 | |
| Raining with high winds | 7119 | 7119 | |
| Raining without high winds | 57054 | 57054 | |
| Snowing with high winds | 733 | 733 | |
| Snowing without high winds | 2707 | 2707 | |
| Unknown | 7668 | 7668 | |



```
accident_current = accident_current[accident_current["Weather_Conditions"] != "Other"]
accident_current = accident_current[accident_current["Weather_Conditions"] != "Unknown"]
print(accident_current.shape)
print(accident_current.isnull().sum(axis=0))
```

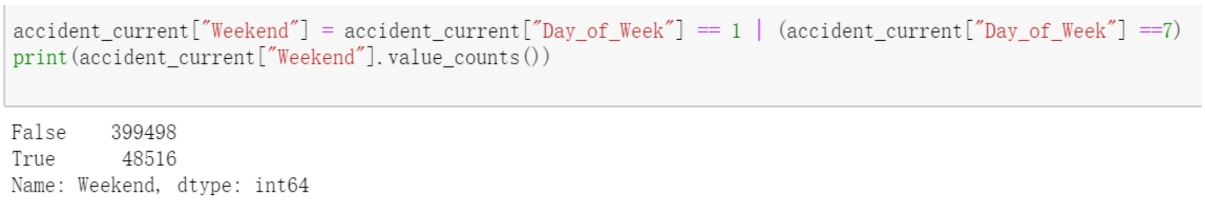
```
(448014, 13)
Police_Force               0
Accident_Severity          0
Number_of_Vehicles         0
Number_of_Casualties       0
Date                       0
Day_of_Week                0
Time                       0
Road_Type                  0
Speed_limit                0
Light_Conditions           0
Weather_Conditions         0
Road_Surface_Conditions    0
Urban_or_Rural_Area        0
dtype: int64
```

Now if we look the missing value again, we have 0% missing rate for 13 attributes

## 3.3 Construct the data

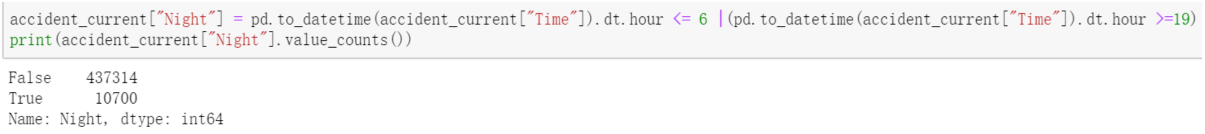
- Deriving attributes

One of the goals of my data mining in this study is to discover the relationship between accident severity and the external conditions and also seeking the pattern of levels of accident severity based on the time, day of the week. There are no outliers and extreme value in my parent data. To seek the time relevant factors that may be related to traffic accident severity. I derive two attributes called “Day or night” and “Weekdays or Weekends”. The first attribute set the time between 19 clocks and 6 clocks as the night, and 7 to 18 as the day. Other attributes divided “Day of Week” filed into two parts, Saturday and Sunday treat as weekends and Monday to Friday treat as Weekdays. Consider the road conditions may be varied among these two attributes, I derive these two attributes for further modeling.



```
accident_current["Weekend"] = accident_current["Day_of_Week"] == 1 | (accident_current["Day_of_Week"] ==7)
print(accident_current["Weekend"].value_counts())
```

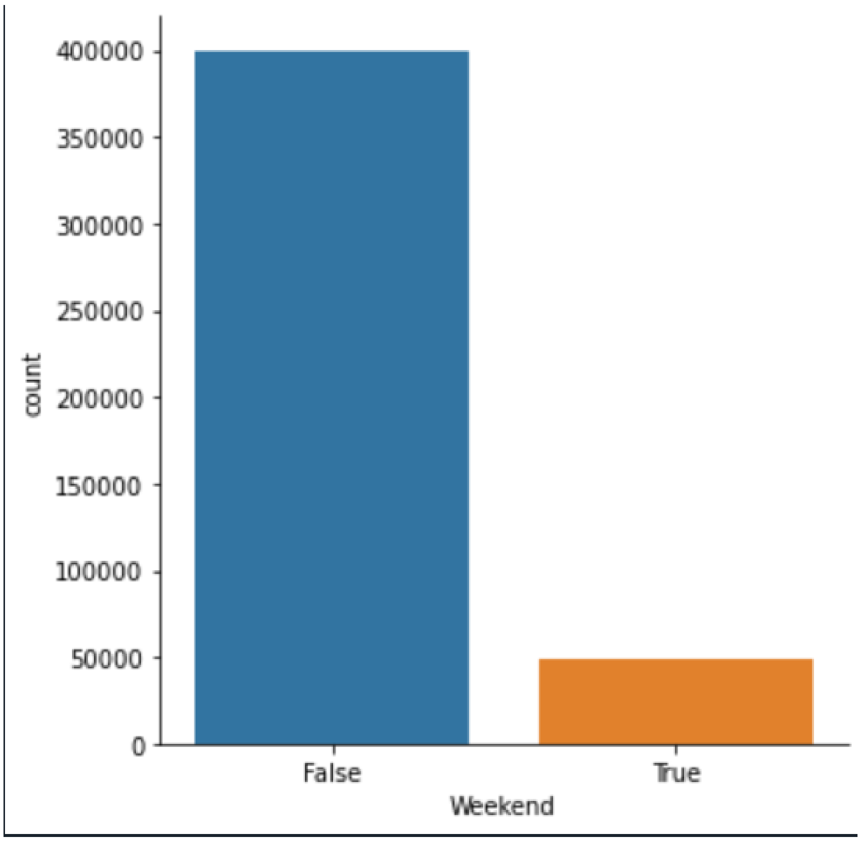
```
False    399498
True      48516
Name: Weekend, dtype: int64
```

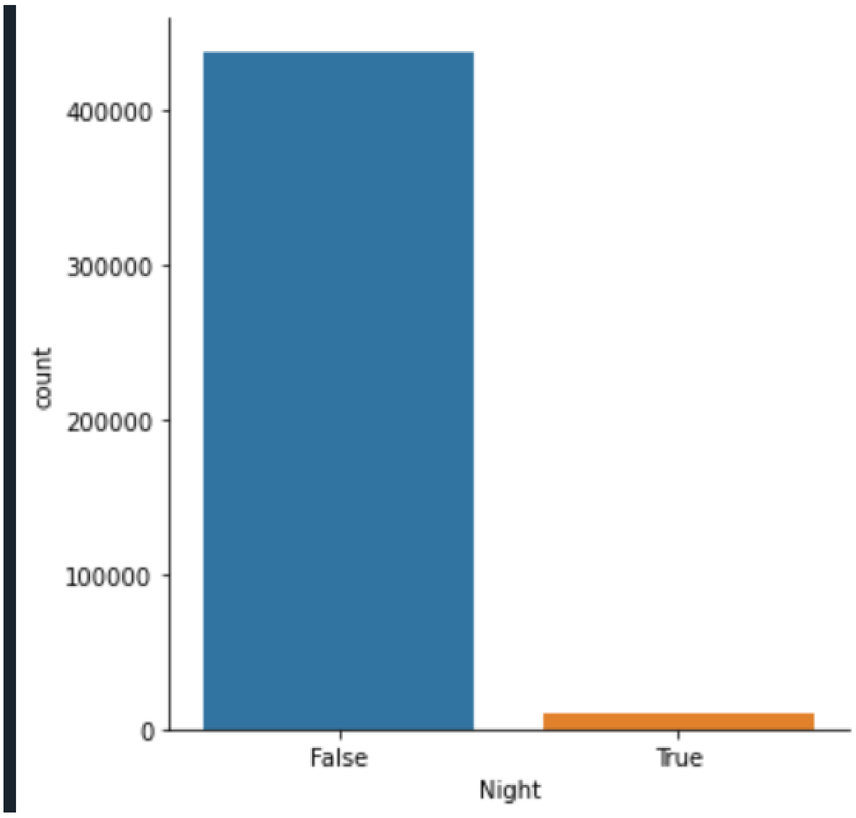


```
accident_current["Night"] = pd.to_datetime(accident_current["Time"]).dt.hour <= 6 |(pd.to_datetime(accident_current["Time"]).dt.hour >=19)
print(accident_current["Night"].value_counts())
```

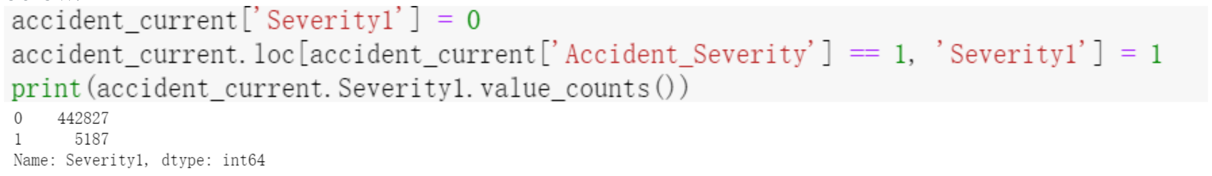
```
False    437314
True      10700
Name: Night, dtype: int64
```

After that, two attributes distributed like below,


400000
350000
300000
250000
200000
150000
100000
50000
0
count
False
True
Weekend


400000
300000
200000
100000
0
count
False
True
Night

For the target value, I derive an attribute named “Severity1” which measures the severity of traffic accidents based on the “Accident_Severity”. The accidents with severity level 1 are much more serious than accidents of other levels, between which the division is far from clear-cut. Therefore, I decided to focus on level 1 accidents and regroup the levels of severity into level 1 versus the other two levels. And the distribution of Severity1 shows below.

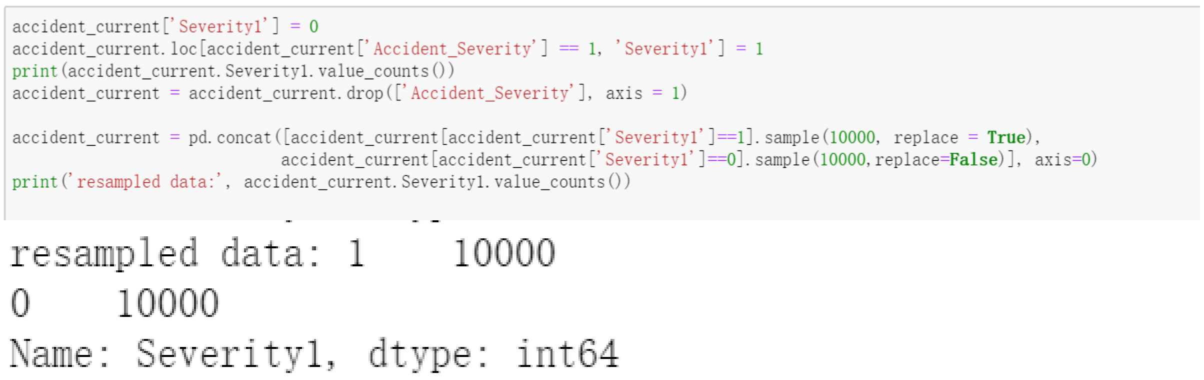


```
accident_current['Severity1'] = 0
accident_current.loc[accident_current['Accident_Severity'] == 1, 'Severity1'] = 1
print(accident_current.Severity1.value_counts())
```

```
0    442827
1      5187
Name: Severity1, dtype: int64
```

- Balance observations

For data analysis purposes, we need to handle the unbalanced data. It may have an unexpected impact on the further prediction and analysis of my model, because my model may classify most of the observations to the major group of target attributes and ignore the minor group. Therefore, I need to balance my data for further analysis. Because the observations of severity level 1 are much less than the others two levels. To address this issue, the combination of over- and under-sampling will be used since the dataset is large enough. level 1 will be randomly oversampled to 10000 and the other two levels will be randomly under-sampled to 10000.



```
accident_current['Severity1'] = 0
accident_current.loc[accident_current['Accident_Severity'] == 1, 'Severity1'] = 1
print(accident_current.Severity1.value_counts())
accident_current = accident_current.drop(['Accident_Severity'], axis = 1)

accident_current = pd.concat([accident_current[accident_current['Severity1']==1].sample(10000, replace = True),
                              accident_current[accident_current['Severity1']==0].sample(10000,replace=False)], axis=0)
print('resampled data:', accident_current.Severity1.value_counts())
```

```
resampled data: 1    10000
0    10000
Name: Severity1, dtype: int64
```

3.4 Integrated data

As I mentioned in step two, the data set used for this study was compiled by a researcher called Dave Fisher-Hickey(https://www.kaggle.com/daveianhickey/2000-16-traffic-flow-england-scotland-wales). Data were collected and summarised by UK officials. The traffic data were gathered at 1.6 million accidents between 2000 and 2016. The raw data can be download in the UK Department of Transport website(https://roadtraffic.dft.gov.uk/). Since the data were completed, I did not find any resources can be supplemented. In the case of any supplemented data set, I can merge with an additional dataset.

3.5 Format the data as required

The data was filtered and selected in previous steps. In general, I Filter out the unnecessary attributes and select the valid records, discard the NULL, and unknown observations for some key factors. Also, I derive a new attribute named “level of vehicle crash involved”, and balance the observations. Therefore, the number of observations was 20000 after sampling data. The inspection of the data after processing shown below,

| | Road_Type | Speed_limit | Light_Conditions | Weather_Conditions | Road_Surface_Conditions | Urban_or_Rural_Area | Weekend | Night | Severity1 |
|---|---|---|---|---|---|---|---|---|---|
| **235930** | Single carriageway | 60 | Darkeness: No street lighting | Fine without high winds | Dry | 1 | 0 | 0 | 1 |
| **348561** | Single carriageway | 30 | Daylight: Street light present | Fine without high winds | Dry | 2 | 1 | 0 | 1 |
| **62223** | Single carriageway | 30 | Darkness: Street lights present and lit | Fine without high winds | Dry | 2 | 0 | 0 | 1 |
| **446182** | Single carriageway | 50 | Daylight: Street light present | Fine without high winds | Dry | 2 | 0 | 0 | 1 |
| **336279** | Dual carriageway | 40 | Darkness: Street lights present and lit | Fine without high winds | Dry | 1 | 0 | 0 | 1 |

## Step 4. Data Transformation

### 4.1 Reduce the data

Feature selection is an essential part of the data analysis process. However, for the data mining goal of this study, I am interested in finding the correlation between attributes. I remove the number of vehicles and the number of casualties since it is highly related to the traffic accident severity, not the external factors that may be related to the accidents. I also remove the day of the week and time for my model, since I already construct the new attributes that are related to time and day of the week. Finally, there are seven predictors left to our model.

```
columns = accident_current.columns.tolist()
columns = [c for c in columns if c not in["Accident_Severity","Police_Force","Number_of_Vehicles",
                                          "Number_of_Casualties","Date","Day_of_Week","Time"]]
target = "Severity1"

accident_current = accident_current[columns]
accident_current.dtypes
```

```
Road_Type                  object
Speed_limit                 int64
Light_Conditions           object
Weather_Conditions         object
Road_Surface_Conditions    object
Urban_or_Rural_Area         int64
Weekend                     int64
Night                       int64
Severity1                   int64
dtype: object
```

### 4.2 Project the data

In statistics, data transformation is to apply a certain mathematical function to each point in the data set, to replace each data point Zi with the converted value Yi = f (Zi), where f is a function. Transformations are often applied to make the data look closer to the assumptions of the statistical inference process to be applied or to improve the interpretability or appearance of the graph. In this study, so far, there is no indication that data conversion is needed, whether it is for the first or second goal [6].

## Step 5 Data-mining Methods Selection

### 5.1 Match and discuss DM methods

- Supervised Learning

Supervised learning refers to the existence of supervisor as teachers. Basically, supervised learning is a kind of learning in which we teach or train machines with well labelled data, which means that some data has been marked as the correct label. Then, a new set of examples (data) is provided to the machine, so that the supervised learning algorithm can analyze the training data (training sample set) and produce correct results from the marked data. Supervised learning can divide into two kinds of algorithms. The first is classification, when the output variable is a category. The regression problem is when the output is an actual value, such as weight [7].

- Unsupervised Learning

Unsupervised learning is the training machines with information that is neither classified nor labelled, and allows algorithms to operate on this information without guidance. The task of the machine is to group the unordered information according to the similarity, pattern and difference, without training the data in advance. Unlike supervised learning, no teachers are provided, which means that the machine will not receive any training before prediction. Therefore, the machine can only find hidden structures in unlabelled data. Unsupervised learning can be divided into two kinds of algorithms. Clustering issues are when you want to discover groups inherent in your data, such as grouping customers based on purchasing behavior. The problem with association rule learning is that you want to find rules that describe most of your data such as people who buy x tend to buy y [7].

According to the data mining objectives in this study:

- To determine and rank the key factors which have the biggest impact on the traffic accident severity in the dataset.
- Try to find the pattern of the traffic accident severity such as time and day of the week.

For the above goals, supervised learning should be used to achieve it, the target filed is “severity” and the data type should be Nominal.

### 5.2 Select the appropriate DM method(s) in a logical manner.

Based on the different data mining methods discussed above, the data mining method chosen in this study is classification and regression under supervised learning.
For the first objective, the goal is to find the relationship between traffic accident severity and other factors that may be related to the severity. In previous steps, a list of potentially relevant attributes is selected, and a classification method will be applied here to model the relationship between the target attribute and other attributes. For the second and the third objectives, the goal is to find and analyze patterns and trends of traffic accident severity.

# Step 6 Data-mining algorithms Selection

## 6.1 Conduct exploratory analysis and discuss

In this step, I will explore data mining algorithms. Based on the data mining objectives of this study, the data mining method will be used in this study includes the regression model and classification in supervised learning. There are a lot of algorithms available under these categories, I will now discuss some of these algorithms and discuss the suitability of them for this study:

- Classification

1. Decision Tree

A decision tree is a flowchart-like structure in which each internal node represents a "test" on an attribute (e.g. whether a coin flip comes up heads or tails), each branch represents the outcome of the test, and each leaf node represents a class label (decision taken after computing all attributes). The paths from the root to leaf represent classification rules (“Decision tree”, n.d.). Decision tree is one of the potential algorithms for analysis regarding the first data mining goal of this study which is to find the relationship between accidents severity and other external factors. Output of a decision tree is interpretable, and the running efficiency is high. However, it is easy to overfit the data [8].

2. Neural Network

A neural network is a network or circuit of neurons, or in a modern sense, an artificial neural network, composed of artificial neurons or nodes. Thus, a neural network is either a biological neural network, made up of real biological neurons or an artificial neural network, for solving artificial intelligence problems. The connections of the biological neuron are modeled as weights. A positive weight reflects an excitatory connection, while negative values mean inhibitory connections. All inputs are modified by a weight and summed. This activity is referred as a linear combination. Finally, an activation function controls the amplitude of the output. For example, an acceptable range of output is usually between 0 and 1, or it could be −1 and 1 (“Neural network”, n.d.) [9].
Neural network is normally used when there is a categorical output variable with multiple levels. Also, the result is not interpretable and the running efficient is low and it normally overfit the data. Therefore, it might not be very suitable for this study.

3. Gradient Boosting Tree

Gradient boosting is a machine learning technique for regression and classification problems, which produces a prediction model in the form of an ensemble of weak prediction models, typically decision trees. It builds the model in a stage-wise fashion like other boosting methods do, and it generalizes them by allowing optimization of an arbitrary differentiable loss function.
Gradient boosting is typically used with decision trees (especially CART trees) of a fixed size as base learners. For this special case, Friedman proposes a modification to gradient boosting method which improves the quality of fit of each base learner.

4. Naïve Bayes

Naive Bayes is a simple technique for constructing classifiers: models that assign class labels to problem instances, represented as vectors of feature values, where the class labels are drawn from some finite set. There is not a single algorithm for training such classifiers,

but a family of algorithms based on a common principle: all naive Bayes classifiers assume that the value of a particular feature is independent of the value of any other feature, given the class variable. For example, a fruit may be considered to be an apple if it is red, round, and about 10 cm in diameter. A naive Bayes classifier considers each of these features to contribute independently to the probability that this fruit is an apple, regardless of any possible correlations between the color, roundness, and diameter features ("Naive Bayes classifier", n.d.)[10] .
Naïve Bayes classifier is normally used in tasks involving Natural Language Processing because this kind of task is normally easy to be converted to a multivariate data format and a Naïve Bayes classifier is fast and easy to construct. Therefore, it might not be very suitable for this study.

- Regression

1. Linear Regression

In statistics, linear regression is a linear approach to modeling the relationship between a scalar response (or dependent variable) and one or more explanatory variables (or independent variables). The case of one explanatory variable is called a simple linear regression. For more than one explanatory variable, the process is called multiple linear regression ("Linear regression", n.d.)[11].
Linear regression could be one of the algorithms for analysis regarding the first data mining goal of this study which is to find the relationship between severity and other external factors. However, it is normally used when the target field is continuous so that the Logistic regression discussed below might be a better choice for this study.

2. Random Forest

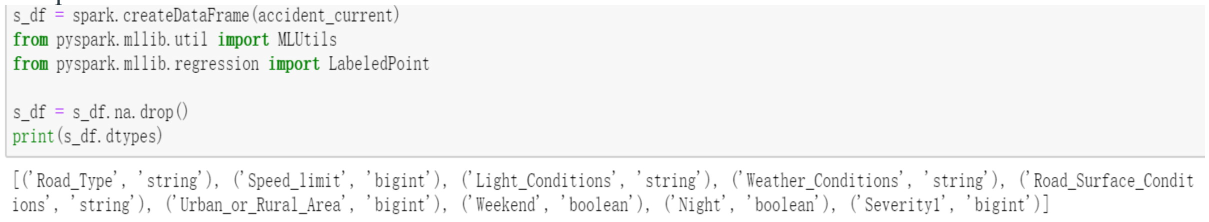
Random forests or random decision forests are an ensemble learning method for classification, regression, and other tasks that operate by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees. Random decision forests correct for decision trees' habit of overfitting to their training set ("Random forest", n.d.)[12].
Logistic regression is the algorithm that best suitable for the first data mining goal of this study in theory because the output variable of the patient data is binary. Also, it is relatively easy to be constructed and the result is interpretable.

- Time Series Analysis

Time series analysis comprises methods for analyzing time-series data to extract meaningful statistics and other characteristics of the data. Time series forecasting is the use of a model to predict future values based on previously observed values. While regression analysis is often employed in such a way as to test theories that the current values of one or more independent time series affect the current value of another time series, this type of analysis of time series is not called "time series analysis", which focuses on comparing values of a single time series or multiple dependent time series at different points in time. Interrupted time series analysis is the analysis of interventions on a single time series ("Time series", n.d.) [13].
Time series analysis is the only algorithm that is appropriated for the goals of the study which is to seek and analyze for time-series patterns.

6.2 Select data-mining algorithms based on exploratory analysis

Based on the data mining objectives and the discussion above, I will choose decision tree algorithms, logistic regression, and random forest as the data mining algorithms for this study. For the first data mining goal of this study, the decision tree algorithm, Logistic regression, and Random forest algorithm will be used in the analysis because they can handle binary output variable type, efficient in running time and the result is easy to interpret.



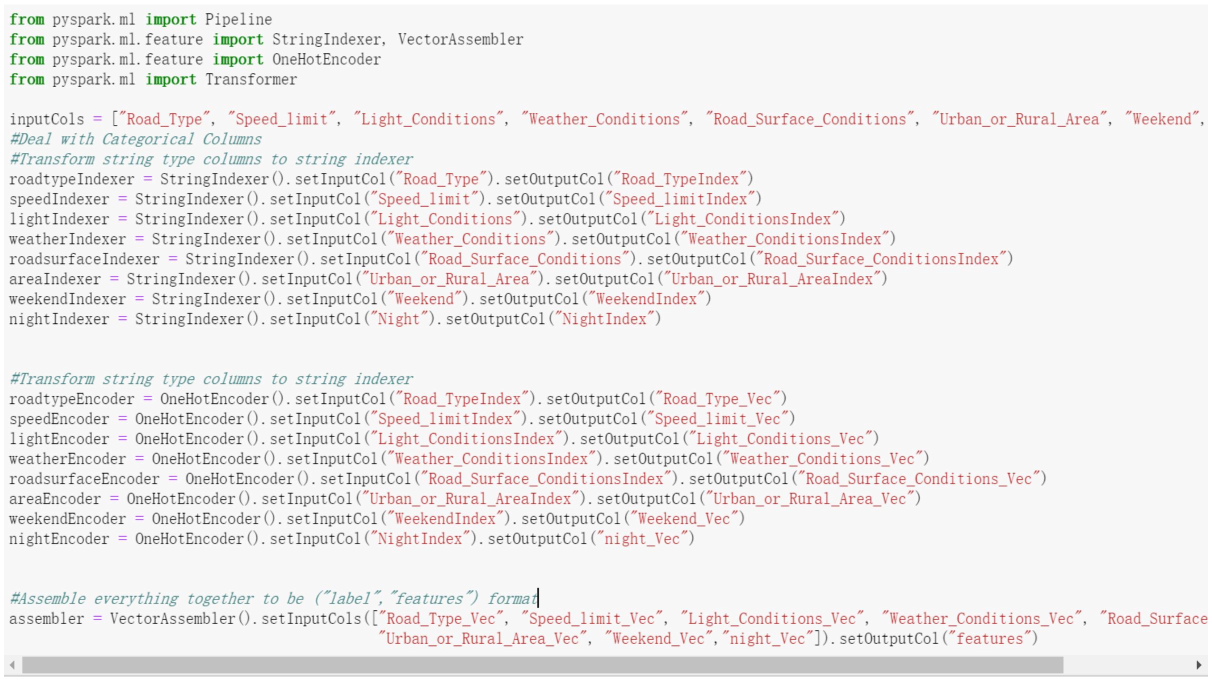
```
s_df = spark.createDataFrame(accident_current)
from pyspark.mllib.util import MLUtils
from pyspark.mllib.regression import LabeledPoint

s_df = s_df.na.drop()
print(s_df.dtypes)
```

```
[('Road_Type', 'string'), ('Speed_limit', 'bigint'), ('Light_Conditions', 'string'), ('Weather_Conditions', 'string'), ('Road_Surface_Condit
ions', 'string'), ('Urban_or_Rural_Area', 'bigint'), ('Weekend', 'boolean'), ('Night', 'boolean'), ('Severity1', 'bigint')]
```

6.3 Construct the appropriate models and choose relevant parameters

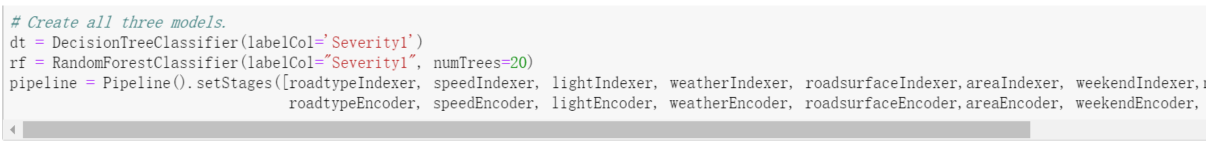
For the first data mining objective which is to find the relationship between accident severity and potential factors that may be related to the severity.

In PySpark, a dataset needs to be transformed as a "libsvm" format to be able to fitted by the tree based classifiers. The following figure shows the way of doing that:

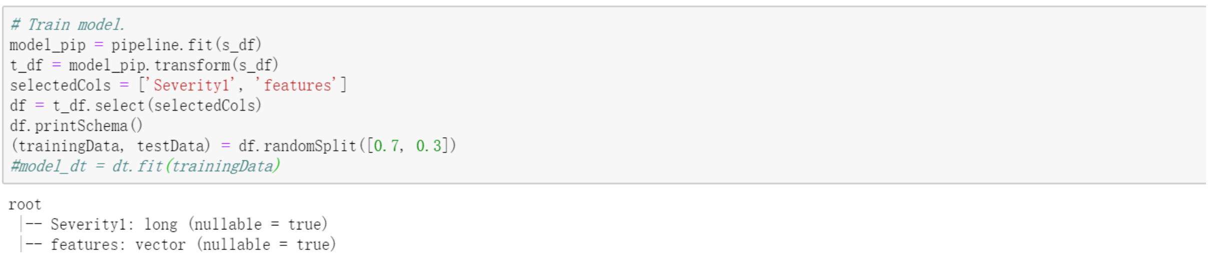


```
from pyspark.ml import Pipeline
from pyspark.ml.feature import StringIndexer, VectorAssembler
from pyspark.ml.feature import OneHotEncoder
from pyspark.ml import Transformer

inputCols = ["Road_Type", "Speed_limit", "Light_Conditions", "Weather_Conditions", "Road_Surface_Conditions", "Urban_or_Rural_Area", "Weekend",
#Deal with Categorical Columns
#Transform string type columns to string indexer
roadtypeIndexer = StringIndexer().setInputCol("Road_Type").setOutputCol("Road_TypeIndex")
speedIndexer = StringIndexer().setInputCol("Speed_limit").setOutputCol("Speed_limitIndex")
lightIndexer = StringIndexer().setInputCol("Light_Conditions").setOutputCol("Light_ConditionsIndex")
weatherIndexer = StringIndexer().setInputCol("Weather_Conditions").setOutputCol("Weather_ConditionsIndex")
roadsurfaceIndexer = StringIndexer().setInputCol("Road_Surface_Conditions").setOutputCol("Road_Surface_ConditionsIndex")
areaIndexer = StringIndexer().setInputCol("Urban_or_Rural_Area").setOutputCol("Urban_or_Rural_AreaIndex")
weekendIndexer = StringIndexer().setInputCol("Weekend").setOutputCol("WeekendIndex")
nightIndexer = StringIndexer().setInputCol("Night").setOutputCol("NightIndex")


#Transform string type columns to string indexer
roadtypeEncoder = OneHotEncoder().setInputCol("Road_TypeIndex").setOutputCol("Road_Type_Vec")
speedEncoder = OneHotEncoder().setInputCol("Speed_limitIndex").setOutputCol("Speed_limit_Vec")
lightEncoder = OneHotEncoder().setInputCol("Light_ConditionsIndex").setOutputCol("Light_Conditions_Vec")
weatherEncoder = OneHotEncoder().setInputCol("Weather_ConditionsIndex").setOutputCol("Weather_Conditions_Vec")
roadsurfaceEncoder = OneHotEncoder().setInputCol("Road_Surface_ConditionsIndex").setOutputCol("Road_Surface_Conditions_Vec")
areaEncoder = OneHotEncoder().setInputCol("Urban_or_Rural_AreaIndex").setOutputCol("Urban_or_Rural_Area_Vec")
weekendEncoder = OneHotEncoder().setInputCol("WeekendIndex").setOutputCol("Weekend_Vec")
nightEncoder = OneHotEncoder().setInputCol("NightIndex").setOutputCol("night_Vec")


#Assemble everything together to be ("label","features") format
assembler = VectorAssembler().setInputCols(["Road_Type_Vec", "Speed_limit_Vec", "Light_Conditions_Vec", "Weather_Conditions_Vec", "Road_Surface
                                            "Urban_or_Rural_Area_Vec", "Weekend_Vec","night_Vec"]).setOutputCol("features")
```

After building the "indexers" and "Encoders", they would be passed in a pipeline model to transform the data:



```
# Create all three models.
dt = DecisionTreeClassifier(labelCol='Severity1')
rf = RandomForestClassifier(labelCol="Severity1", numTrees=20)
pipeline = Pipeline().setStages([roadtypeIndexer, speedIndexer, lightIndexer, weatherIndexer, roadsurfaceIndexer,areaIndexer, weekendIndexer,
                                roadtypeEncoder, speedEncoder, lightEncoder, weatherEncoder, roadsurfaceEncoder,areaEncoder, weekendEncoder,
```

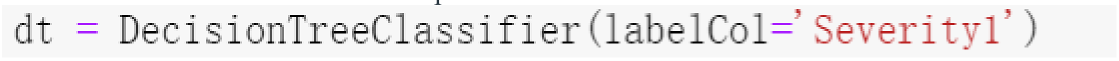


```
# Train model.
model_pip = pipeline.fit(s_df)
t_df = model_pip.transform(s_df)
selectedCols = ['Severity1', 'features']
df = t_df.select(selectedCols)
df.printSchema()
(trainingData, testData) = df.randomSplit([0.7, 0.3])
#model_dt = dt.fit(trainingData)
```

```
root
 |-- Severity1: long (nullable = true)
 |-- features: vector (nullable = true)
```

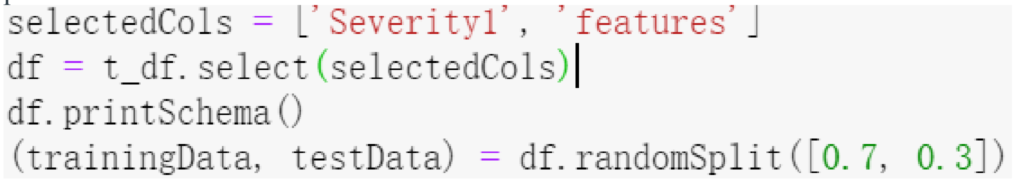
- Decision Tree

The Decision Tree model setting for the first data mining goal is shown below. The model also able to calculate the predictor's importance which is one of the advantages of the Decision Tree algorithm.  I set the number of trees to 20, the more trees you have, the more computation time. But this could also significantly increase accuracy. So there's a tradeoff. Also, I set in the Type node, the target variable is “Severity1” which including two levels, 1 for True and 0 for False. The first step should be bulid the decision tree classifier like below:



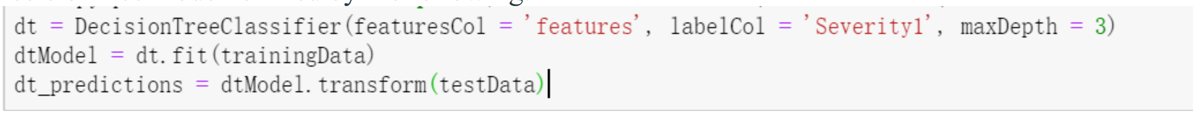
```
dt = DecisionTreeClassifier(labelCol='Severity1')
```

To avoid overfitting the data, across-validation with 70% training set and 30% testing set is used. We will be using a training set to fit the model and a testing set to produce a prediction.



```
selectedCols = ['Severity1', 'features']
df = t_df.select(selectedCols)
df.printSchema()
(trainingData, testData) = df.randomSplit([0.7, 0.3])
```

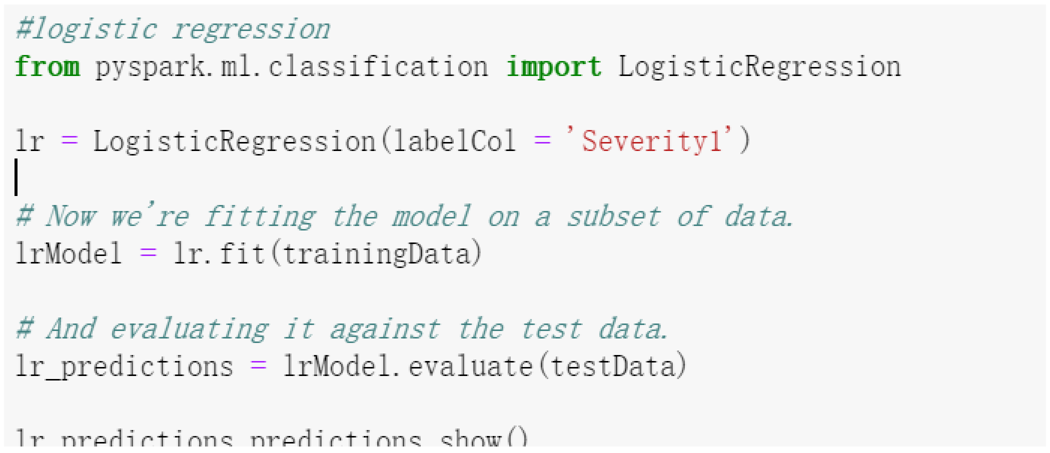
Decision tree model is fitted by the following:



```
dt = DecisionTreeClassifier(featuresCol = 'features', labelCol = 'Severity1', maxDepth = 3)
dtModel = dt.fit(trainingData)
dt_predictions = dtModel.transform(testData)
```

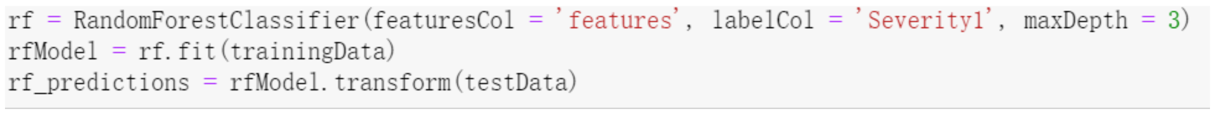
- Logistics regression

The Logistic regression model coding for the data mining goal is shown below. I have set up the model to calculate the under Receiver operating characteristic area of the model that we can make compassion with the other models.



```
#logistic regression
from pyspark.ml.classification import LogisticRegression

lr = LogisticRegression(labelCol = 'Severity1')

# Now we're fitting the model on a subset of data.
lrModel = lr.fit(trainingData)

# And evaluating it against the test data.
lr_predictions = lrModel.evaluate(testData)

lr_predictions.predictions.show()
```

- Random Forest



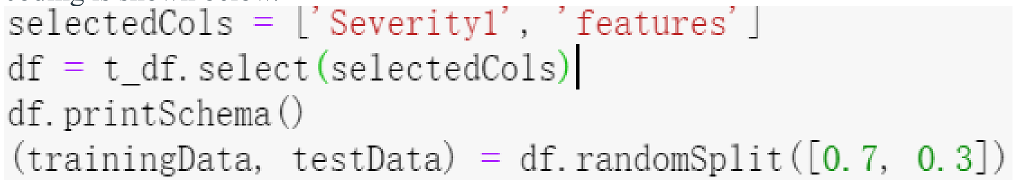
```
rf = RandomForestClassifier(featuresCol = 'features', labelCol = 'Severity1', maxDepth = 3)
rfModel = rf.fit(trainingData)
rf_predictions = rfModel.transform(testData)
```

The maxiumu depth of random forest is set to 3. A deeper tree can fit more complicated functions. Therefore, increasing tree depth should increase performance on the training set. But, increased flexibility also gives greater ability to overfit the data, and generalization performance may suffer if depth is increased too far.

## Step 7 Data Mining

### 7.1 Create and justify test designs

We don't want overfitting or under fitting happen, because they affect the predictability of our model . The choice of the train/test split and cross validation helps to avoid these problems.
As I mentioned above, to avoid overfitting to the data, a cross-validation test is designed to split the data into a 70/30 radio training and testing data for the analysis of the data mining goal of this study. This can be done by using the "df.randomSplit" in Pyspark, and the coding is shown below.

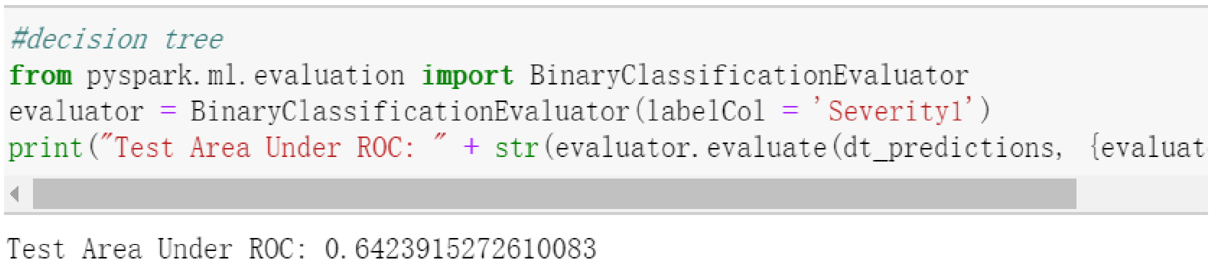


```
selectedCols = ['Severity1', 'features']
df = t_df.select(selectedCols)
df.printSchema()
(trainingData, testData) = df.randomSplit([0.7, 0.3])
```

### 7.2 Conduct data mining – classify, regress, cluster

For the data mining objective which is to find the relationship between accident severity and other external factors:

- Decision tree

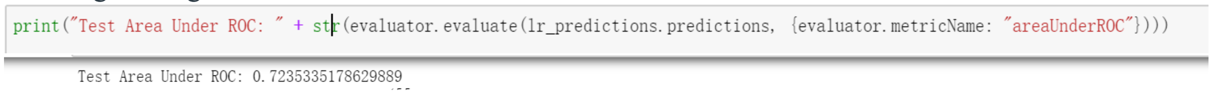


```
#decision tree
from pyspark.ml.evaluation import BinaryClassificationEvaluator
evaluator = BinaryClassificationEvaluator(labelCol = 'Severity1')
print("Test Area Under ROC: " + str(evaluator.evaluate(dt_predictions, {evaluat
```

```
Test Area Under ROC: 0.6423915272610083
```

The model achieved around 64% accuracy on both training and testing data. We will discuss and try to explain later.

- Logistic regression

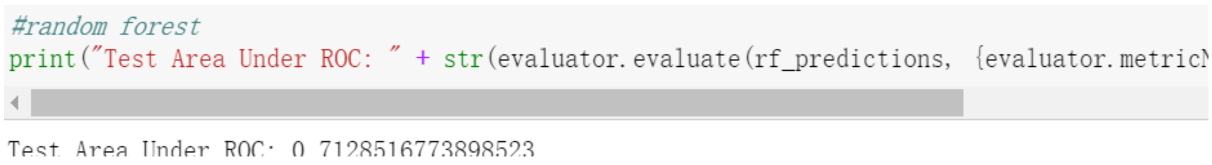


```
print("Test Area Under ROC: " + str(evaluator.evaluate(lr_predictions.predictions, {evaluator.metricName: "areaUnderROC"})))
```

```
Test Area Under ROC: 0.7235335178629889
```

The result of logistic regression achieved over 72% accuracy on the test data. Out of three model, logestic regression seems to outperform the other two.

- Random Forest



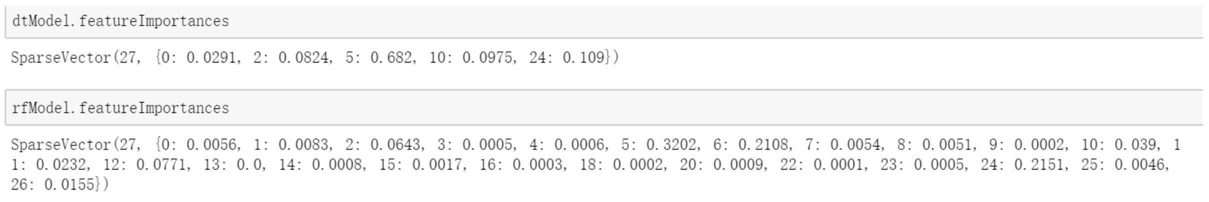
```
#random forest
print("Test Area Under ROC: " + str(evaluator.evaluate(rf_predictions, {evaluator.metricN
```

```
Test Area Under ROC: 0.7128516773898523
```

Random Forest achieved similar accuracy with the decision tree algorithm which is around 71%.

7.3 Search for pattern

For the data mining objective which is to find the relationship between accident severity and other external potential factors that may relate it. The comparison from the two analyses results of the Decision tree and Random forest model is below:



```
dtModel.featureImportances
```

```
SparseVector(27, {0: 0.0291, 2: 0.0824, 5: 0.682, 10: 0.0975, 24: 0.109})
```

```
rfModel.featureImportances
```

```
SparseVector(27, {0: 0.0056, 1: 0.0083, 2: 0.0643, 3: 0.0005, 4: 0.0006, 5: 0.3202, 6: 0.2108, 7: 0.0054, 8: 0.0051, 9: 0.0002, 10: 0.039, 1
1: 0.0232, 12: 0.0771, 13: 0.0, 14: 0.0008, 15: 0.0017, 16: 0.0003, 18: 0.0002, 20: 0.0009, 22: 0.0001, 23: 0.0005, 24: 0.2151, 25: 0.0046,
26: 0.0155})
```

One of the drawbacks for PySpark is that tree based model like Decision tree model cannot be easily viewed like Sklearn.

## Step 8 Interpretation

### 8.1 Study and discuss the mined patterns

Choosing the right model type is one of the key factors for successful data mining. In the previous steps, the random forest model and Decision tree model showed that they are worse than the logistic regression model for analyzing the data mining goal of this study, which is to find the relationship between the traffic accident severity and other external factors. However, the accuracy of these two models is unsatisfied. This is far from the end of these data mining goals, and there is still much room for improvement. For example, as I mentioned earlier, the data on traffic accidents have supplementary information from a third party. Once there is suitable supplementary data in the future that can implement with the data set, I currently use, the quality of this research can be improved.

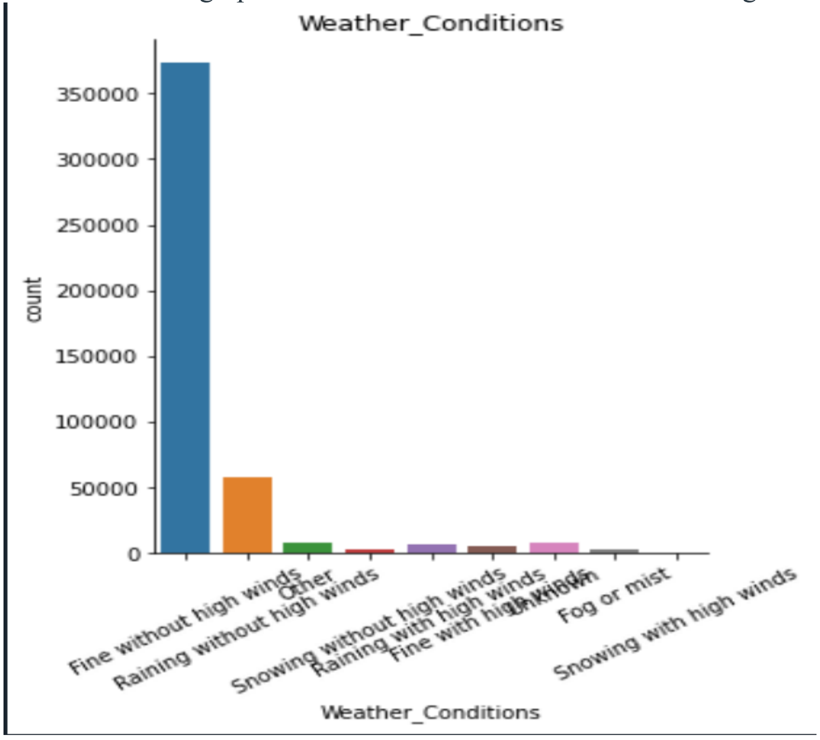
8.2 Visualize the data, result, models, and patterns

Data visualization is another important step in data mining. Generally, after visualizing the data, you can get some important information, such as whether the data needs to be transformed or how the overall trend of the data changes. For the first data mining objective which is to find the relationship between the traffic accident severity and other external factors.

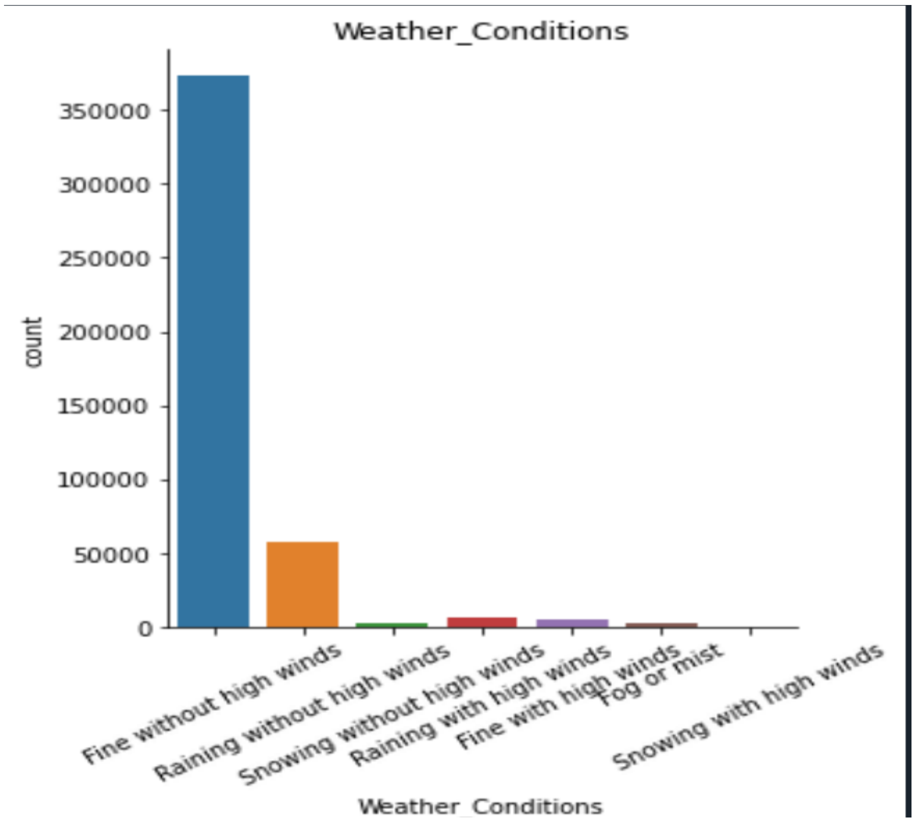
- Attribute distribution

A good way to understand the data is to visualize the attributes. In the previous steps, if you visualize the attribute distribution in advance, it is much easier to clean up the data. Besides, data visualization provides a very intuitive way to know the impact of data cleansing on the distribution of this attribute.

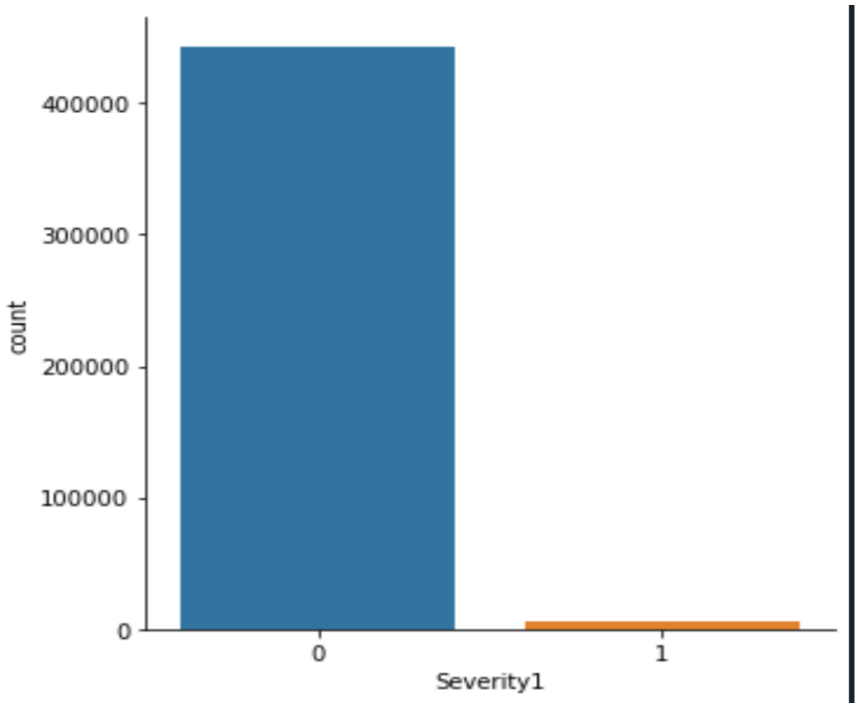
The distribution graph for “Weather conditions” before cleaning:



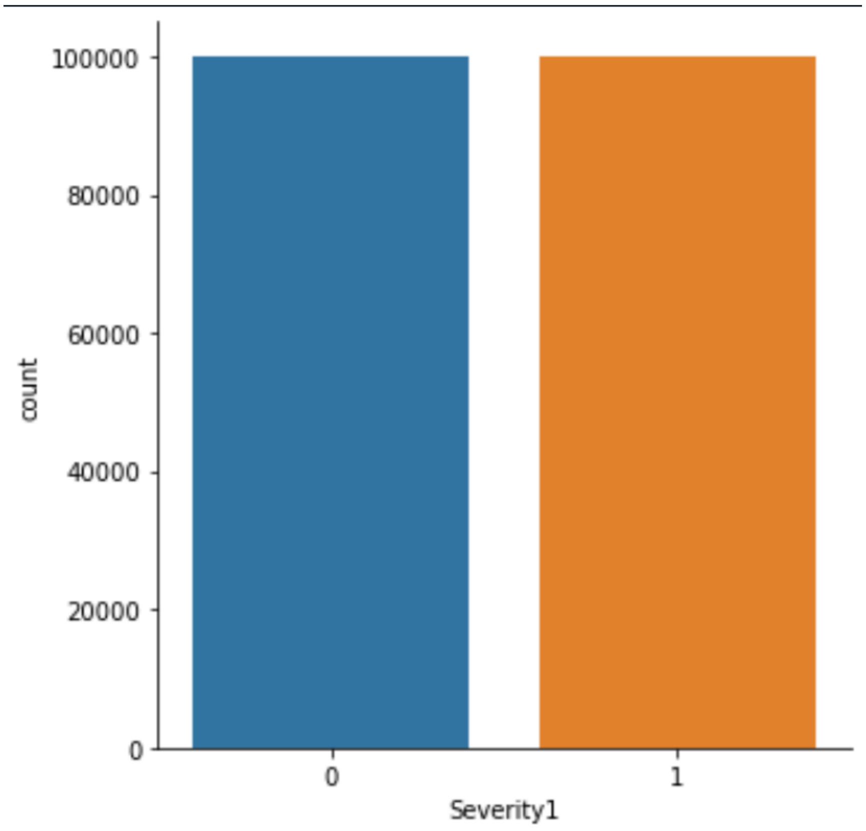
The distribution graph for “Weather conditions” after data cleaning:



The distribution graph for target attribute "Severity1" before sampling:

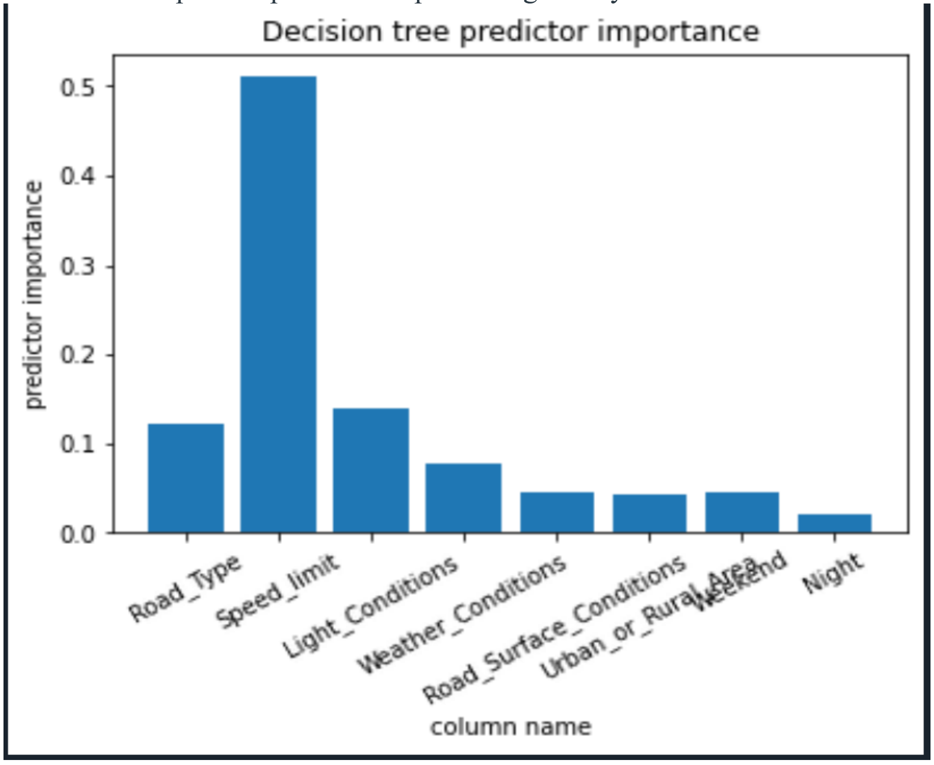


The distribution graph for target attribute "Severity1" after sampling:

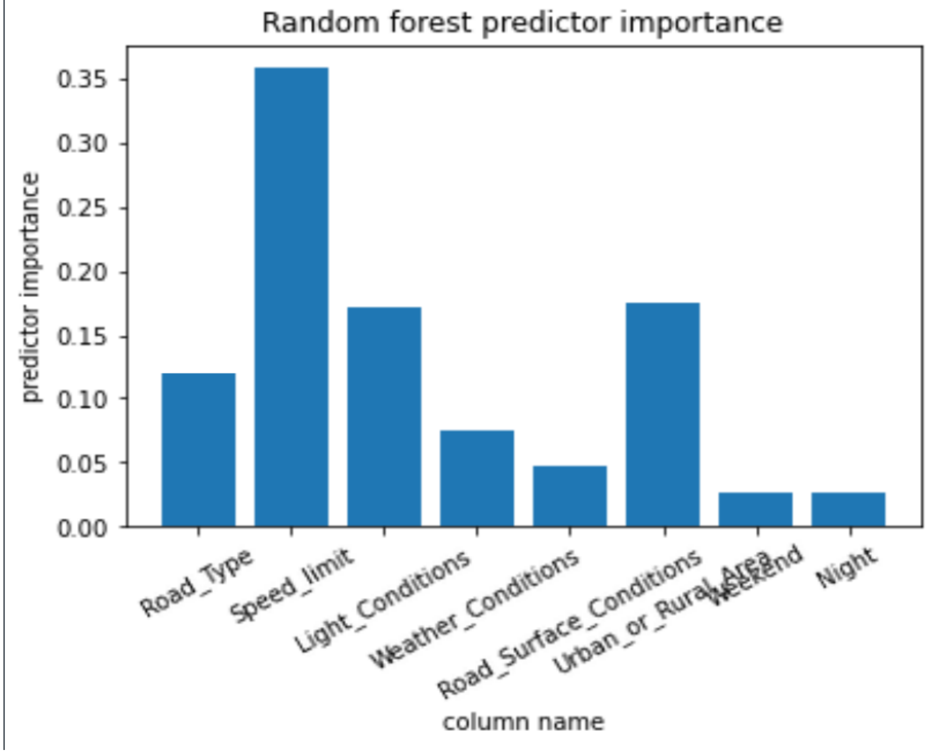


- Predictor importance

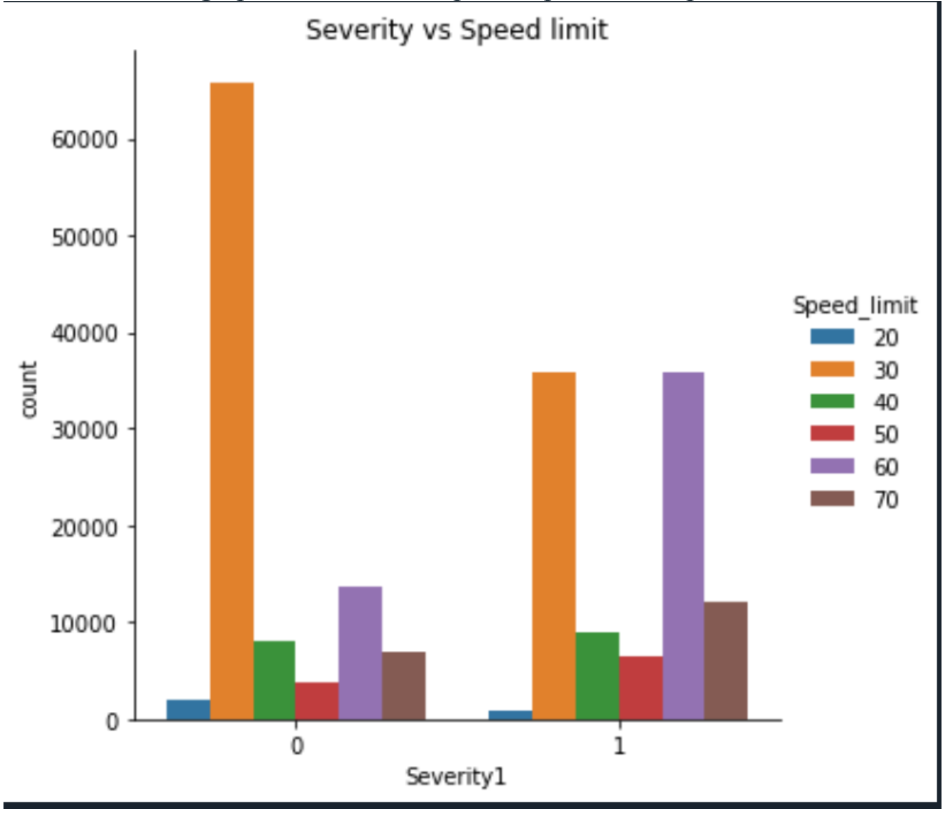
As I mentioned earlier, the importance of predictors calculated by two models is relatively close. The bar plot for predictor importance given by the decision tree model:



The bar plot for predictor importance given by random forest model:



The distribution graph for the most important predictor "Speed limit" and Severity:



- Model result

As mentioned earlier, only the decision tree model could give out an intuitive way to visualize the result. The graph of the decision tree structure is too wide, so I would not put it here.

8.3 Interpret the results, models and patterns

From the result of the decision tree model, it is very difficult to directly interpret the results because the generated decision tree is too large and takes a long time to understand deeply. On the other hand, since this is one of the advantages of decision tree algorithms, it is more straightforward to explain and discuss the importance of predictors from the model. The overall accuracy of both training and testing data is disappointed. This could explain by car accidents mainly caused by the chance and subjective factors such as the inattention inattentive driver. Some external factors may explain, but it is not the decisive factor. For example, bad light conditions and bad weather tend to cause more car accidents. The results show that the important predictors are speed limit, light conditions, and weather conditions which is consistent with common sense. It will have more serious car accidents and more car crashes on the highway or motorway, but the minor accident in the urban road. Because the urban road has a restricted speed limit, also a good lighting system, and road surface conditions. The distribution of speed limits can validate it, more serious car accidents above 60 km/h speed limit road. However, the result shows that the weekend is an insignificant variable. Therefore, the traffic flow may not be associated with the accident severity.

8.4 Assess and evaluate the result, model, and pattern

After the entire data mining procedures, the results can be said to be imperfect. The study aimed to find the relationship between accident severity and external factors that may be related to it and the result is clear even though the direct interpretation is hard. The result was modeled by the decision tree and random forest, and we check the predictor importance, accuracy of prediction, and other evaluation presented by either table or graph. However, as stated before, there is still a lot of improvement that can be made.

8.5 Multiple iterations

1. Business understanding
   I read lots of news and articles about car accidents. Regarding the data mining goals of this study, severity may also be related to other factors such as its vehicle type, etc. This may be the direction that this study can be extended.
2. Data understanding
   Repeat this step for double-checking the data.
3. Data preparation
   Repeat this step for double-checking the data.
4. Data transformation
   Repeating this step may not make much sense for this study.
5. Data mining method selection
   Repeating this step may not make much sense for this study.
6. Data mining algorithm selection
   In the previous steps, I used the decision tree and random forest algorithm to model. I think that the model using the gradient boosting algorithm is also very worth trying because the gradient boosting is a machine learning technique for regression and classification, which predures a prediction model in the form of an ensemble of weak prediction model, and it often gives out a good accuracy.

6.3 Construct the appropriate models and choose relevant parameters

- Gradient boosting

```
from pyspark.ml.classification import GBTClassifier
from pyspark.ml.evaluation import MulticlassClassificationEvaluator

gbt = GBTClassifier(labelCol='Severity1',featuresCol='features')
gbt_model = gbt.fit(trainingData)

gbt_predictions = gbt_model.transform(testData)
```

7. Data mining
   7.2 Conduct data mining - classify, regress, cluster

- Gradient boosting

```
my_binary_gbt_eval = BinaryClassificationEvaluator(labelCol='Severity1', rawPredictionCol='prediction')
print("GBT")
print(my_binary_gbt_eval.evaluate(gbt_predictions))
```

```
GBT
0.6778993149984056
```

The result from the gradient boosting model is worse than the result of the decision tree and random forest model. The model achieved 68% accuracy on the test data. The reason of that might be related to the hyperparameter settings or data types of the dataset used.

8. Interpretation
   8.1 Study and discuss the mined pattern
   Although gradient boosting algorithm are generally regarded as algorithms that can provide more accurate results, the above results do not support this view. One reason may be because the data used in this study is more categorical variables, so it seems to be more suitable for decision tree algorithm or random forest algorithm.

   8.2 Interpret the results, models, and patterns
   One of the biggest disadvantages of gradient boosting algorithm is the inability to interpret the results because we don't understand the mechanism within the ensemble. This is one of the reasons why I think it is not suitable for this study.

## Disclaimer

"I acknowledge that the submitted work is my own original work in accordance with the University of Auckland guidelines and policies on academic integrity and copyright. (See: https://www.auckland.ac.nz/en/students/forms-policies-and-guidelines/student-policies-and-guidelines/academic-integrity-copyright.htmlLinks to an external site.).

## Reference

1. Health – United Nations Sustainable Development. (n.d.). Retrieved from https://www.un.org/sustainabledevelopment/health/
2. Road Traffic Injuries & Deaths: A Global Problem. (2019, December 18). Retrieved from https://www.cdc.gov/injury/features/global-road-safety/index.html
3. (2018, December 18). Retrieved from https://www.ibm.com/support/pages/spss-modeler-182-documentation
4. Road traffic injuries. (n.d.). Retrieved from https://www.who.int/health-topics/road-safety#tab=tab_1
5. Road to Zero: A New Road Safety Strategy for NZ. (n.d.). Retrieved from https://www.transport.govt.nz/multi-modal/keystrategiesandplans/road-safety-strategy/
6. Data transformation (statistics). (2020, July 15). Retrieved from https://en.wikipedia.org/wiki/Data_transformation_(statistics)
7. Supervised and Unsupervised learning. (2019, October 01). Retrieved from https://www.geeksforgeeks.org/supervised-unsupervised-learning/
8. Decision tree. (2020, May 29). Retrieved from https://en.wikipedia.org/wiki/Decision_tree
9. Naive Bayes classifier. (2020, August 26). Retrieved from https://en.wikipedia.org/wiki/Naive_Bayes_classifier
10. Neural network. (2020, August 03). Retrieved from https://en.wikipedia.org/wiki/Neural_network
11. Linear regression. (2020, August 17). Retrieved from https://en.wikipedia.org/wiki/Linear_regression .
12. Random forest. (2020, August 11). Retrieved from https://en.wikipedia.org/wiki/Random_forest
13. Time series. (2020, August 26). Retrieved from https://en.wikipedia.org/wiki/Time_series