Starting on the IBM Data Science Capstone Report: Car Accident Severity.

Grading criteria:

For this week, you will be required to submit the following:

1. A description of the problem and a discussion of the background. (15 marks)
2. A description of the data and how it will be used to solve the problem. (15 marks)

Business Understanding:

Description of the problem:

In order to reduce the frequency of car crashes in the community, an algorithm must be developed to predict accident severity given the current weather, road and light conditions.

The main goal of this study is to prevent accidents when conditions are bad, that's why this model is meant to alert drivers to remind them to be a little bit more careful, to switch roads or postpone their car rides.

There is another possible targeted audience for this study, the local police, health institutes, insurance companies etc. They can make good use of this model to know when to be fully ready to receive bad news about a specific road, and more importantly take prevention measures to avoid accidents on certain ones.

Discussion of the background:

In most cases, carelessness while driving, using drugs and alcohol or driving too fast are some of the main causes of accidents that can be avoided by implementing stronger regulations.

Besides the above reasons, weather, visibility or road conditions are the major uncontrollable factors which can be avoided by uncovering patterns hidden in the data and declaring a warning to local government, police and drivers on the roads. targeted routes, or alerting the drivers before the road trips.

Data Understanding:

Source of the data:

The data used in this project is the example dataset provided in the course. Including all types of collisions.

It concerns the city of Seattle, WA. It is provided by Seattle Police Dept. and recorded by Traffic Records, with a timeframe of: 2004 to Present.

Data Set Summary

Data Set Basics	
Title	Collisions—All Years
Abstract	All collisions provided by SPD and recorded by Traffic Records.
Description	This includes all types of collisions. Collisions will display at the intersection or mid-block of a segment. Timeframe: 2004 to Present.
Supplemental Information	
Update Frequency	Weekly
Keyword(s)	SDOT, Seattle, Transportation, Accidents, Bicycle, Car, Collisions, Pedestrian, Traffic, Vehicle

Description of the data:

The data consists of 37 independent variables and 194,673 rows. The dependent variable, “SEVERITYCODE”, contains numbers that correspond to different levels of severity caused by an accident from 0 to 4.

Severity codes are as follows:

0: Little to no Probability (Clear Conditions)
1: Very Low Probability — Chance or Property Damage
2: Low Probability — Chance of Injury
3: Mild Probability — Chance of Serious Injury
4: High Probability — Chance of Fatality

Furthermore, because of the existence of a huge unbalance in some attributes occurrences and the existence of null values in some records, the data needs to be preprocessed, cleaned and balanced before any further processing.

How the data will be used to solve the problem:

We have to select the most important features to weigh the severity of accidents in Seattle. Among all the features, the following features have the most influence in the accuracy of the predictions:

The **'WEATHER', 'ROADCOND'** and **'LIGHTCOND'** attributes.

Preprocessing:

Let's count the missing values of the attribute columns that we are using to weigh the severity of the collisions we're going to study.

```
WEATHER          5081
ROADCOND         5012
LIGHTCOND        5170
```

Right now, the columns are of type object, let's count their values:

```
print(df['WEATHER'].value_counts())
```

```
Clear                       111135
Raining                      33145
Overcast                     27714
Unknown                      15091
Snowing                        907
Other                          832
Fog/Smog/Smoke                 569
Sleet/Hail/Freezing Rain       113
Blowing Sand/Dirt               56
Severe Crosswind                25
Partly Cloudy                    5
Name: WEATHER, dtype: int64
```

```
print(df['ROADCOND'].value_counts())
```

```
Dry             124510
Wet              47474
Unknown          15078
Ice               1209
Snow/Slush        1004
Other              132
Standing Water     115
Sand/Mud/Dirt       75
Oil                 64
Name: ROADCOND, dtype: int64
```

```
print(df['LIGHTCOND'].value_counts())
```

```
Daylight                    116137
Dark - Street Lights On      48507
Unknown                      13473
Dusk                          5902
Dawn                          2502
Dark - No Street Lights       1537
Dark - Street Lights Off      1199
Other                          235
Dark - Unknown Lighting         11
Name: LIGHTCOND, dtype: int64
```

Let’s study the balance of our dataset, let’s check if the samples in the SEVERITYCODE column, have nearly equal value count.

```
print(df['SEVERITYCODE'].value_counts())
```

```
1    136485
2     58188
Name: SEVERITYCODE, dtype: int64
```

The class 1 is nearly 3 times the number of rows of class 2, let’s fix the unbalance in the SEVERITYCODE column by down sampling the data to only 58188 of each class.

```
from sklearn.utils import resample

df_maj = df[df.SEVERITYCODE == 1]
df_min = df[df.SEVERITYCODE == 2]

df_sample = resample(df_maj, replace=False, n_samples=58188, random_state=123)
df = pd.concat([df_sample, df_min])

df['SEVERITYCODE'].value_counts()
```

```
]: 2    58188
   1    58188
   Name: SEVERITYCODE, dtype: int64
```

In order to work with our features as categorical values, we should change the type of the columns, and we will add 3 other columns containing the label encoding of categories in the 3 column attributes.

```
WEATHER         113560 non-null category
ROADCOND        113612 non-null category
LIGHTCOND       113528 non-null category
```

Defining then our Feature DataFrame that we will work with for the rest of the study

```
df["WEATHER_c"] = df["WEATHER"].cat.codes
df["ROADCOND_c"] = df["ROADCOND"].cat.codes
df["LIGHTCOND_c"] = df["LIGHTCOND"].cat.codes
Feature = df[['WEATHER','ROADCOND','LIGHTCOND','WEATHER_c','ROADCOND_c','LIGHTCOND_c']]
X = Feature
X.head()
```

]:

	WEATHER	ROADCOND	LIGHTCOND	WEATHER_c	ROADCOND_c	LIGHTCOND_c
65794	Unknown	Unknown	Daylight	10	7	5
56870	Clear	Wet	Dark - Street Lights On	1	8	2
36864	Overcast	Dry	Daylight	4	0	5
29557	Clear	Dry	Daylight	1	0	5
38852	Clear	Dry	Daylight	1	0	5

Let’s now define our labels array.

```
y = df['SEVERITYCODE'].values
y[0:]
```

]: array([1, 1, 1, ..., 2, 2, 2])