DATA

We have the data of road accident of Seattle city, which consists of 1,94,673 cases and have 37 attributes or columns for all these accidents. In our data, the target label is “SEVERITYCODE” where the code for severity of accident is 1 for property damage collision and 2 for Injury collision. We have 37 columns in our data set where many of the columns in our data set are not fit for analysis like how many pedestrians, bicycles, vehicles or individuals involved in accident. There are various columns which are purely for administrative records. We exclude these unnecessary data and then make a bar plot of possible attributes in our data set to see which attributes gives us insight about the road accident severity. Data, in its original form, has many attributes which are unnecessary. First we remove the unnecessary columns.

- Removing Unnecessary Columns

In [6]:
```
#drop unrequired columns

df1 = df.drop(["X","Y","OBJECTID","INCKEY","COLDETKEY","REPORTNO","STATUS","INTKEY","LOCATION","EXCEPTRSNCODE",
               "EXCEPTRSNDESC","SEVERITYDESC","SDOT_COLCODE","SDOT_COLDESC","SDOTCOLNUM","SDOTCOLNUM","ST_COLCODE",
               "ST_COLDESC","SEGLANEKEY","CROSSWALKKEY"],axis=1)
```

In [7]:
```
# drop duplicate column

df1.drop("SEVERITYCODE.1", axis=1, inplace=True)
```

In [8]:
```
df1.head()
```

Out[8]:

	SEVERITYCODE	ADDRTYPE	COLLISIONTYPE	PERSONCOUNT	PEDCOUNT	PEDCYLCOUNT	VEHCOUNT	INCDATE	INCDTTM	JUNCTIONTYPE	INAT
0	2	Intersection	Angles	2	0	0	2	2013-03-27 00:00:00+00:00	2013-03-27 14:54:00	At Intersection (intersection related)	
1	1	Block	Sideswipe	2	0	0	2	2006-12-20 00:00:00+00:00	2006-12-20 18:55:00	Mid-Block (not related to intersection)	
2	1	Block	Parked Car	4	0	0	3	2004-11-18 00:00:00+00:00	2004-11-18 10:20:00	Mid-Block (not related to intersection)	
3	1	Block	Other	3	0	0	3	2013-03-29 00:00:00+00:00	2013-03-29 09:26:00	Mid-Block (not related to intersection)	
4	2	Intersection	Angles	2	0	0	2	2004-01-28 00:00:00+00:00	2004-01-28 08:04:00	At Intersection (intersection related)	

Figure 1

Original data had various missing values which must be replaced or removed. We either replaced them with mostly occurring variables of that column, or completed removed the corresponding rows from the data set.

- Managing Missing Values

In [10]:
```
# replace missing values with N for No
# replace 0 with N for No and 1 with Y for Yes

df1["INATTENTIONIND"].replace(np.nan,"N",inplace=True)
df1["UNDERINFL"].replace("0","N",inplace=True)
df1["UNDERINFL"].replace("1","Y",inplace=True)
df1["SPEEDING"].replace(np.nan,"N",inplace=True)
df1["PEDROWNOTGRNT"].replace(np.nan,"N",inplace=True)

df1.head()
```

Out[10]:

	SEVERITYCODE	ADDRTYPE	COLLISIONTYPE	PERSONCOUNT	PEDCOUNT	PEDCYLCOUNT	VEHCOUNT	INCDATE	INCDTTM	JUNCTIONTYPE	INAT
0	2	Intersection	Angles	2	0	0	2	2013-03-27 00:00:00+00:00	2013-03-27 14:54:00	At Intersection (intersection related)	
1	1	Block	Sideswipe	2	0	0	2	2006-12-20 00:00:00+00:00	2006-12-20 18:55:00	Mid-Block (not related to intersection)	
2	1	Block	Parked Car	4	0	0	3	2004-11-18 00:00:00+00:00	2004-11-18 10:20:00	Mid-Block (not related to intersection)	
3	1	Block	Other	3	0	0	3	2013-03-29 00:00:00+00:00	2013-03-29 09:26:00	Mid-Block (not related to intersection)	
4	2	Intersection	Angles	2	0	0	2	2004-01-28 00:00:00+00:00	2004-01-28 08:04:00	At Intersection (intersection related)	

Figure 2

After plotting bar graphs we saw that data is highly unbalanced, where severity code in class 1 is almost three times the size of class 2. We balanced the data by down sampling it.

- Balancing the Data Set

In [17]:
```
#balancing the sample

from sklearn.utils import resample

majsc = df2[df2["SEVERITYCODE"]==1]
minsc = df2[df2["SEVERITYCODE"]==2]

bal_majsc = resample(majsc, replace = False, n_samples = 56625, random_state=123)

bal_df = pd.concat([bal_majsc, minsc])
bal_df["SEVERITYCODE"].value_counts().to_frame()
```

Out[17]:

	SEVERITYCODE
2	56625
1	56625

Figure 3

We made a bar plot of attributes like speeding, under influence of alcohol, and inattention of driver to study the accidents due to errors on part of individual.

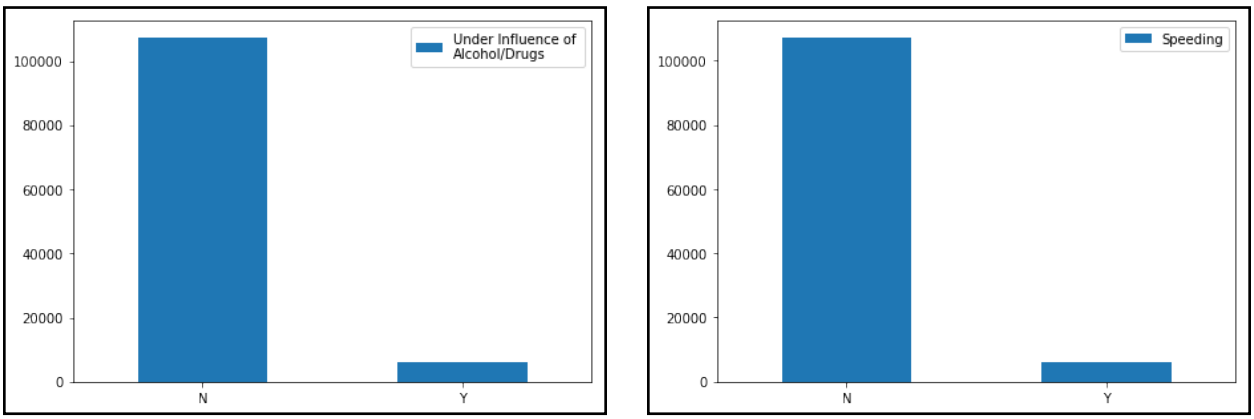


Figure 4

We also checked if the day of the week resembles with chances of accident and its severity. We noticed that working days have a higher frequency of collisions.

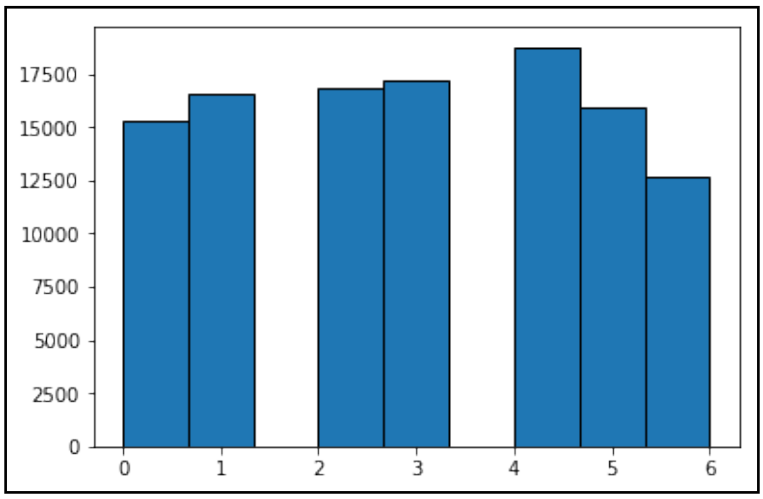
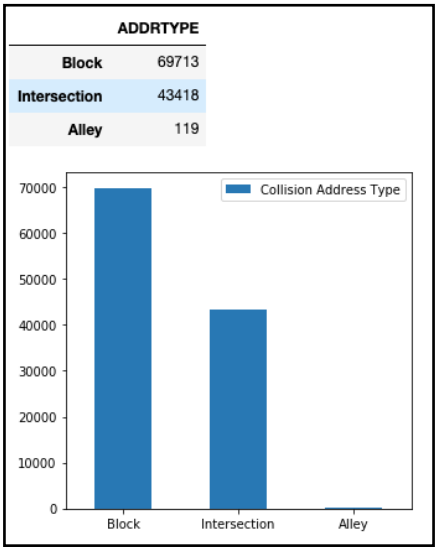


Figure 5

Attributes in our data set are mostly of string types so we used label encoding to make our data suitable for fitting with model. A sample of attribute “ADDRTYPE” is shown here.




```
In [29]: bal_df['ADDRTYPE'].replace(to_replace=['Block','Intersection','Alley'], value=[0,1,2],inplace=True)
         bal_df.head()
```

Out[29]:

	SEVERITYCODE	ADDRTYPE	COLLISIONTYPE	PERSONCOUNT	PEDCOUNT	PEDCYLCOUNT	VEHCOUNT	INCDATE	INCDTTM
129091	1	0	Sideswipe	2	0	0	2	2014-05-17 00:00:00+00:00	2014-05-17 15:35:00
175353	1	1	Left Turn	5	0	0	2	2018-03-11 00:00:00+00:00	2018-03-11 15:04:00
110094	1	0	Parked Car	2	0	0	2	2012-08-26 00:00:00+00:00	2012-08-26 03:55:00
46167	1	1	Left Turn	5	0	0	2	2007-03-14 00:00:00+00:00	2007-03-14 18:00:00
38310	1	0	Parked Car	2	0	0	2	2006-10-19 00:00:00+00:00	2006-10-19 08:38:00

Figure 6

After all this preprocessing we finally select our attributes for model fitting which have significant impact on severity of accident and which can be used for model fitting. The attributes are:

1. **WEATHER**: It reaves about weather conditions of the day
2. **ROADCOND**: It gives information about road conditions.
3. **LIGHTCOND**: It gives information about the kind of light available during accident.
4. **ADDRTYPE**: It reveals the location or address of accident out of Alley, Block and Intersection.

- Selecting the Primary Features for Further Data Analysis and Model Building/Evaluation

In [36]:
```
#defining dependent variable

feat = bal_df[['ROADCOND','LIGHTCOND','WEATHER','ADDRTYPE']]
feat.head()
```

Out[36]:

	ROADCOND	LIGHTCOND	WEATHER	ADDRTYPE
129091	0	0	0	0
175353	0	0	0	1
110094	0	1	0	0
46167	0	0	0	1
38310	1	0	2	0

Figure 7

We got a perfectly balanced dataset with our chosen attributes to determine the target.