CAPSTONE PROJECT FOR DATA SCIENCE

PREDICTING THE SEVERITY OF A ROAD COLLISION BASED ON HISTORICAL DATA OF SEATTLE CITY

SEPTEMBER 2020

RUDRA SHEKHAR

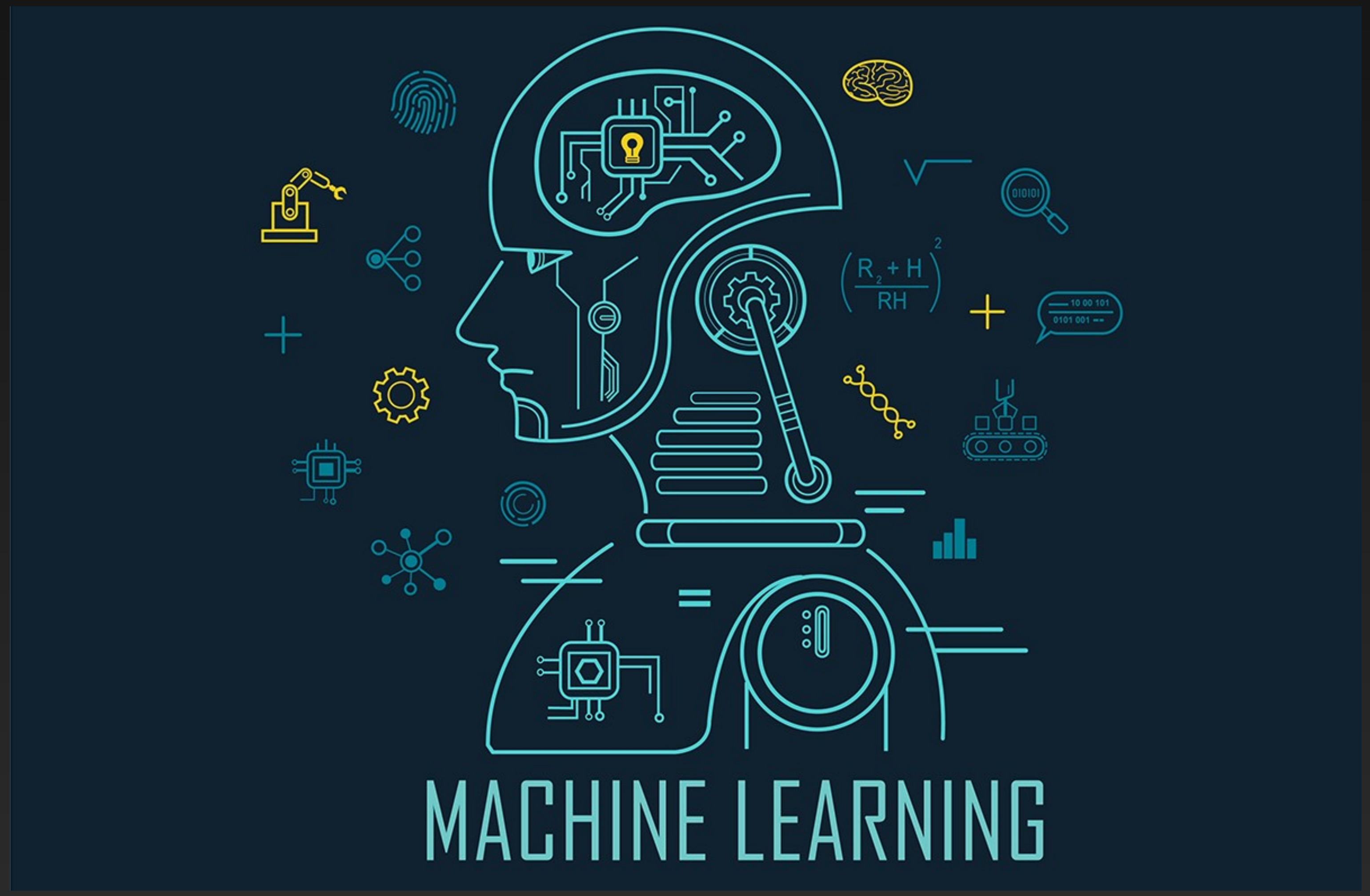
INTRODUCTION

- Road Accidents - the most unnoticed catastrophe
- Can be of varying severity from damage to property to massive loss of human lives.
- Broad Reasons:
  - ❖ Associated with individuals: Over-speeding, drug abuse, lack of attention, etc.
  - ❖ External Factors: Environmental, Infrastructural, etc.

- Ways to control:
  - ❖ Human related factors can be regulated by proactive government legislations.
  - ❖ Infrastructural issues can be handled by administration.
  - ❖ Environmental issues can be taken care of by being attentive.

INTRODUCTION

- Analysis of historical data can predict the role of different factors in road accidents.
- Helpful for different stakeholders:
    - ❖ Government Bodies
    - ❖ Insurance Companies
    - ❖ Common public
- Machine learning algorithms can be deployed to predict the severity and factors responsible for road collisions.



AIM OF THE PROJECT

- Analyze historical data of road accidents in Seattle.
- Use different Machine Learning algorithms to build a predictive model.
- Check the accuracy of the model using accuracy estimators.
- The model can be deployed to predict the severity of a road collision based on various internal & external factors.
- Agencies can use such models to mitigate the impact of road collisions.

SAMPLE & DATA

- Initial sample set consisted of 37 attributes for each incident covering almost 1,94,000 cases.
- Target label is SEVERITYCODE with two values: 1 for Property Damage & 2 for Injury.
- Data needs to be wrangled and pre-processed before it can be used to create any models.

	SEVERITYCODE	X	Y	OBJECTID	INCKEY	COLDETKEY	REPORTNO	STATUS	ADDRTYPE	INTKEY	...	ROADCOND	LIGHTCOND	PEDI
0	2	-122.323148	47.703140	1	1307	1307	3502005	Matched	Intersection	37475.0	...	Wet	Daylight	
1	1	-122.347294	47.647172	2	52200	52200	2607959	Matched	Block	NaN	...	Wet	Dark - Street Lights On	
2	1	-122.334540	47.607871	3	26700	26700	1482393	Matched	Block	NaN	...	Dry	Daylight	
3	1	-122.334803	47.604803	4	1144	1144	3503937	Matched	Block	NaN	...	Dry	Daylight	
4	2	-122.306426	47.545739	5	17700	17700	1807429	Matched	Intersection	34387.0	...	Wet	Daylight	

DATA PRE-PROCESSING

- Removing unnecessary columns

- Replacing/Removing missing values

- Removing Unnecessary Columns

In [6]:
```python
#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]:
```python
# drop duplicate column

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

In [8]:
```python
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)	

- Managing Missing Values

In [10]:
```python
# 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)	

DATA PRE-PROCESSING

- Balancing the data set

- Balancing the Data Set

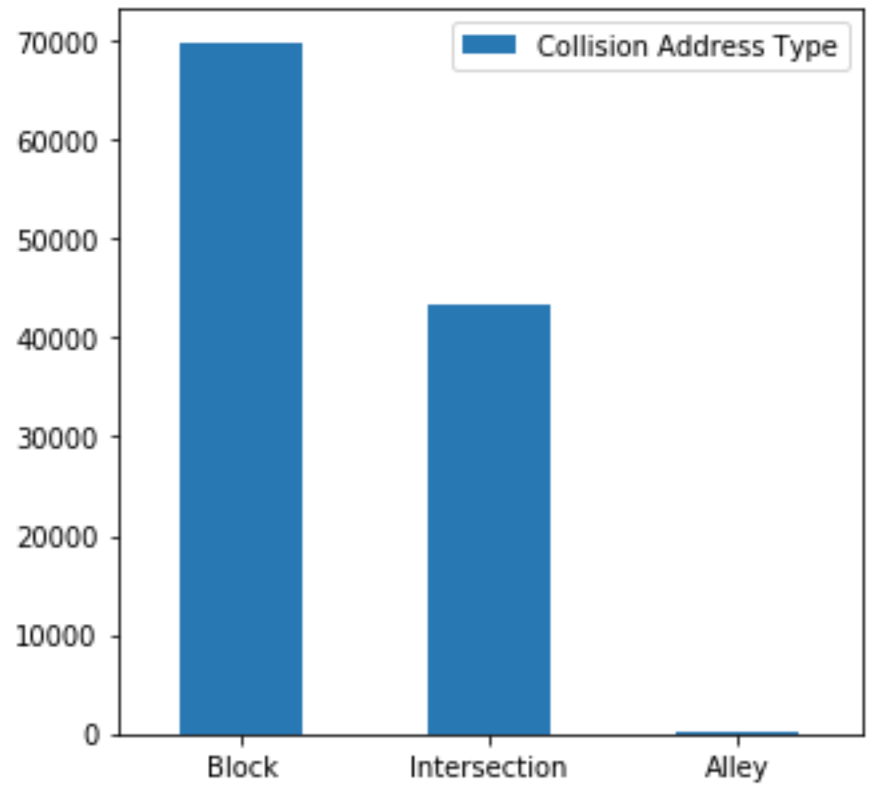
In [17]:
```python
#balancing the sample

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

- Encoding of categorical variables

	ADDRTYPE
Block	69713
Intersection	43418
Alley	119

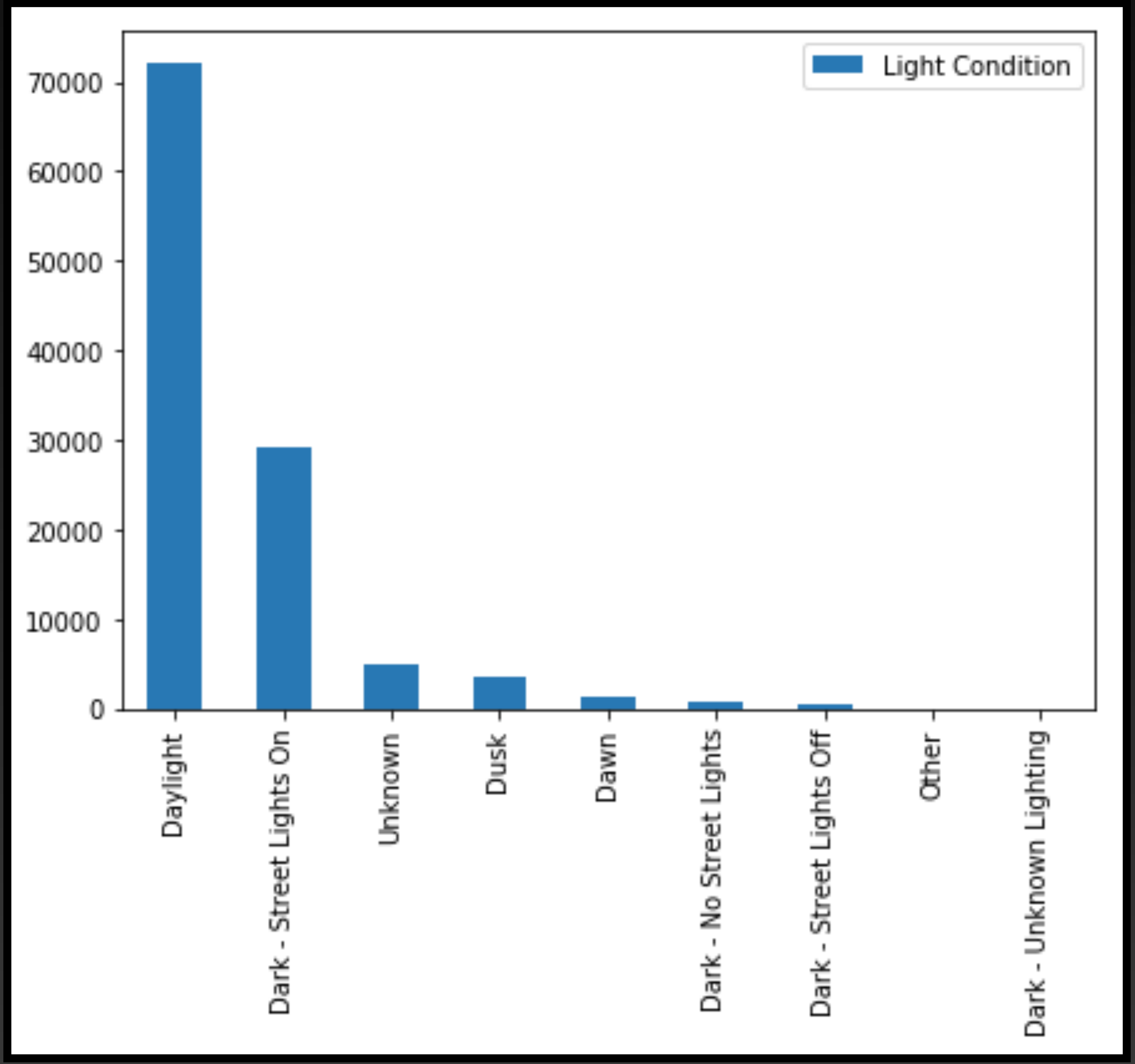



```python
bal_df['ADDRTYPE'].replace(to_replace=['Block','Intersection','Alley'], value=[0,1,2],inplace=True)
bal_df.head()
```

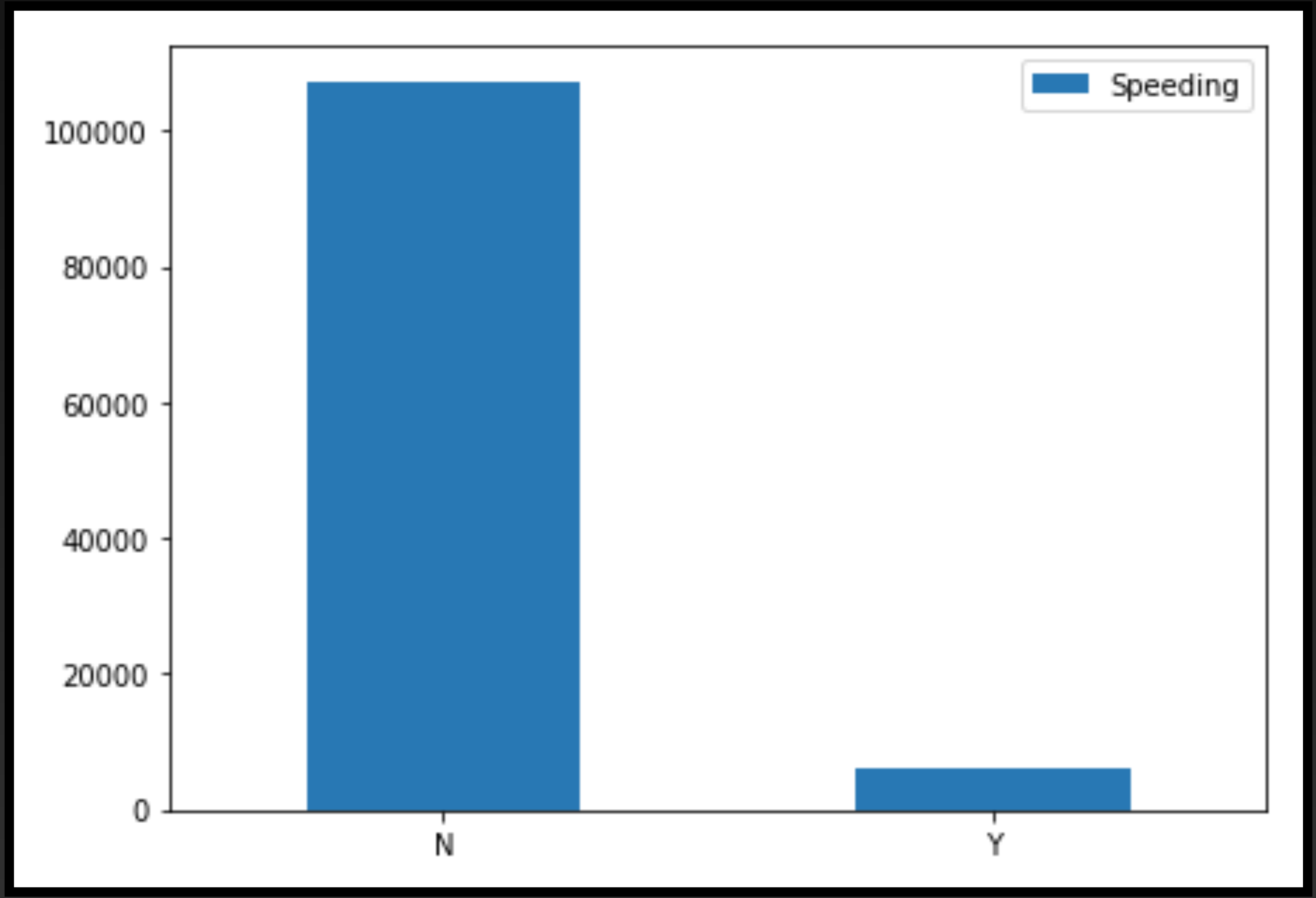
	SEVERITYCODE	ADDRTYPE	COLLISIONTYPE	PERSONCOUNT	PEDCOUNT	PEDCYLCOUNT	VEHCOUNT	INCDATE	INCDTTM
129091	1	0	Sideswipe	2	0	0	2	2014-05-17 00:00:00+00:00	2014-05-17 15:35:00
175353	1	1	Left Turn	5	0	0	2	2018-03-11 00:00:00+00:00	2018-03-11 15:04:00
110094	1	0	Parked Car	2	0	0	2	2012-08-26 00:00:00+00:00	2012-08-26 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

DATA VISUALIZATION

External Factor



Internal Factor



PRIMARY FEATURES FOR DATA ANALYSIS

- Road Condition: Dry, wet, ice-covered, etc.
- Lighting Condition: Daylight, dark with/without street lights, dusk, etc.
- Weather: Rainy, clear, overcast, winds, etc.
- Location: Alley, intersection, block

- 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

METHODOLOGY

- Data normalization

- Normalizing Data

```
In [40]: X = preprocessing.StandardScaler().fit(feat).transform(feat)
         X[0:5]

/Users/kriti/anaconda3/lib/python3.7/site-packages/sklearn/pre
h input dtype int64 were all converted to float64 by StandardS
  return self.partial_fit(X, y)
/Users/kriti/anaconda3/lib/python3.7/site-packages/ipykernel_l
ype int64 were all converted to float64 by StandardScaler.
  """Entry point for launching an IPython kernel.

Out[40]: array([[-0.57976317, -0.56088911, -0.66811458, -0.78852984],
                [-0.57976317, -0.56088911, -0.66811458,  1.25703101],
                [-0.57976317,  0.40466171, -0.66811458, -0.78852984],
                [-0.57976317, -0.56088911, -0.66811458,  1.25703101],
                [ 0.84254862, -0.56088911,  1.25683458, -0.78852984]])
```

- Data splitting into train and test set

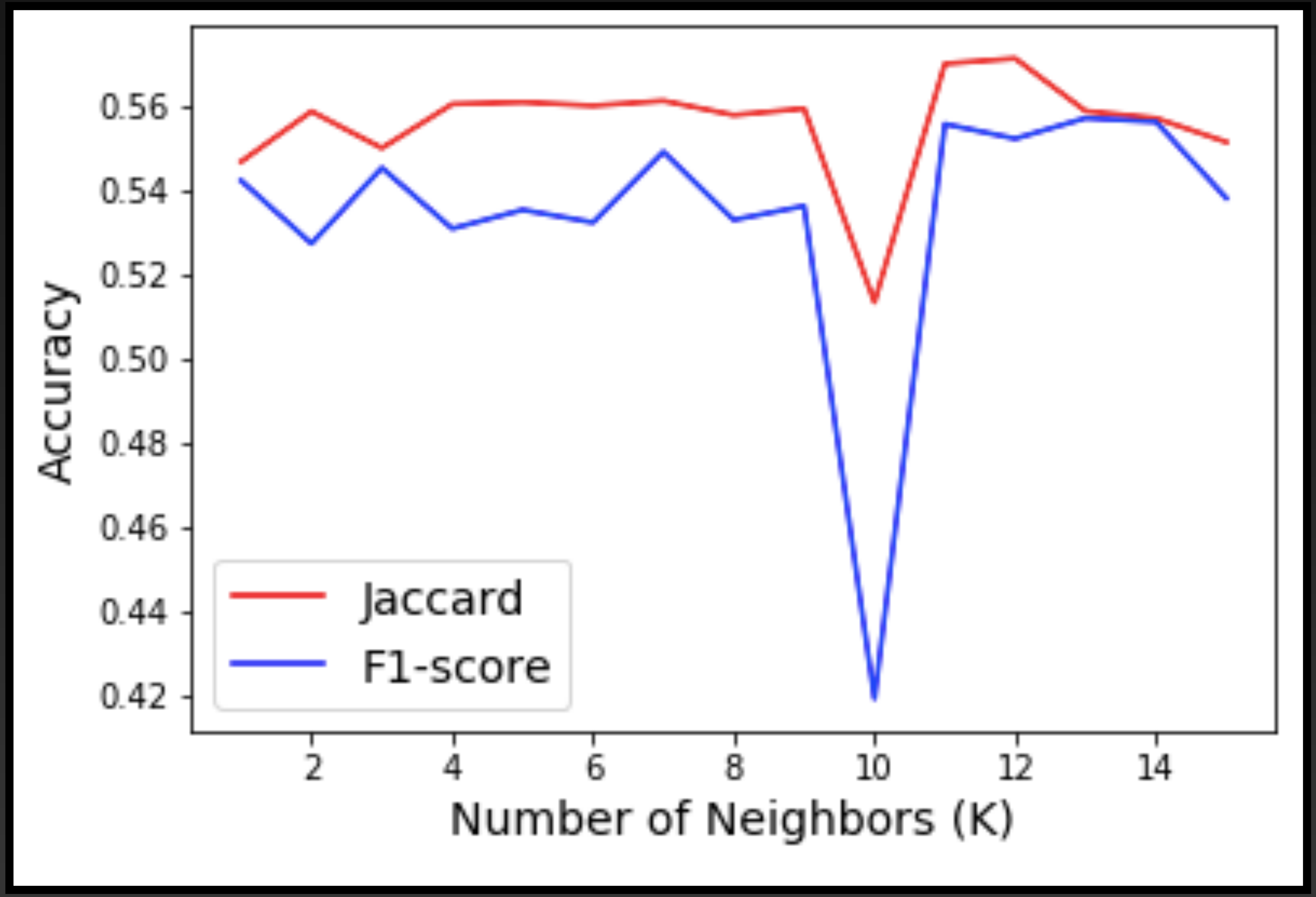
- Splitting Data into Train and Test Set

```
In [41]: from sklearn.model_selection import train_test_split

         x_train, x_test, y_train, y_test = train_test_split(X, Y, test_size=0.25, random_state=4)

         print ('Train Dataset:', x_train.shape,  y_train.shape)
         print ('Test Dataset:', x_test.shape,  y_test.shape)

Train Dataset: (84937, 4) (84937,)
Test Dataset: (28313, 4) (28313,)
```

MODEL BUILDING

K Nearest Neighbor (KNN)

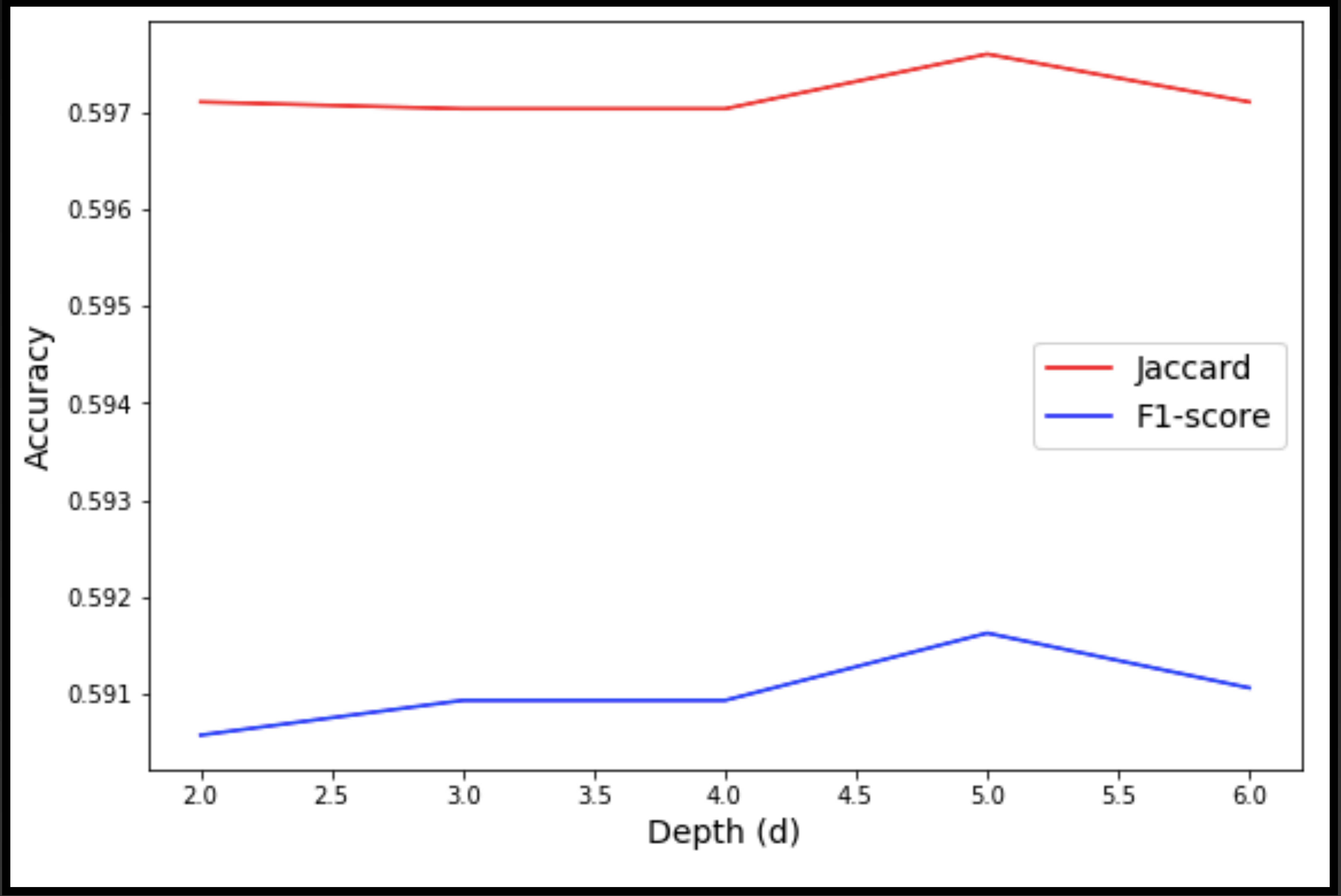
- Varied value of K from 1 to 15.
- Used Jaccard Index to determine the best accuracy model.
- Trained the model (K = 12) with train data.
- Evaluated the model using test data.



MODEL BUILDING

Decision Tree

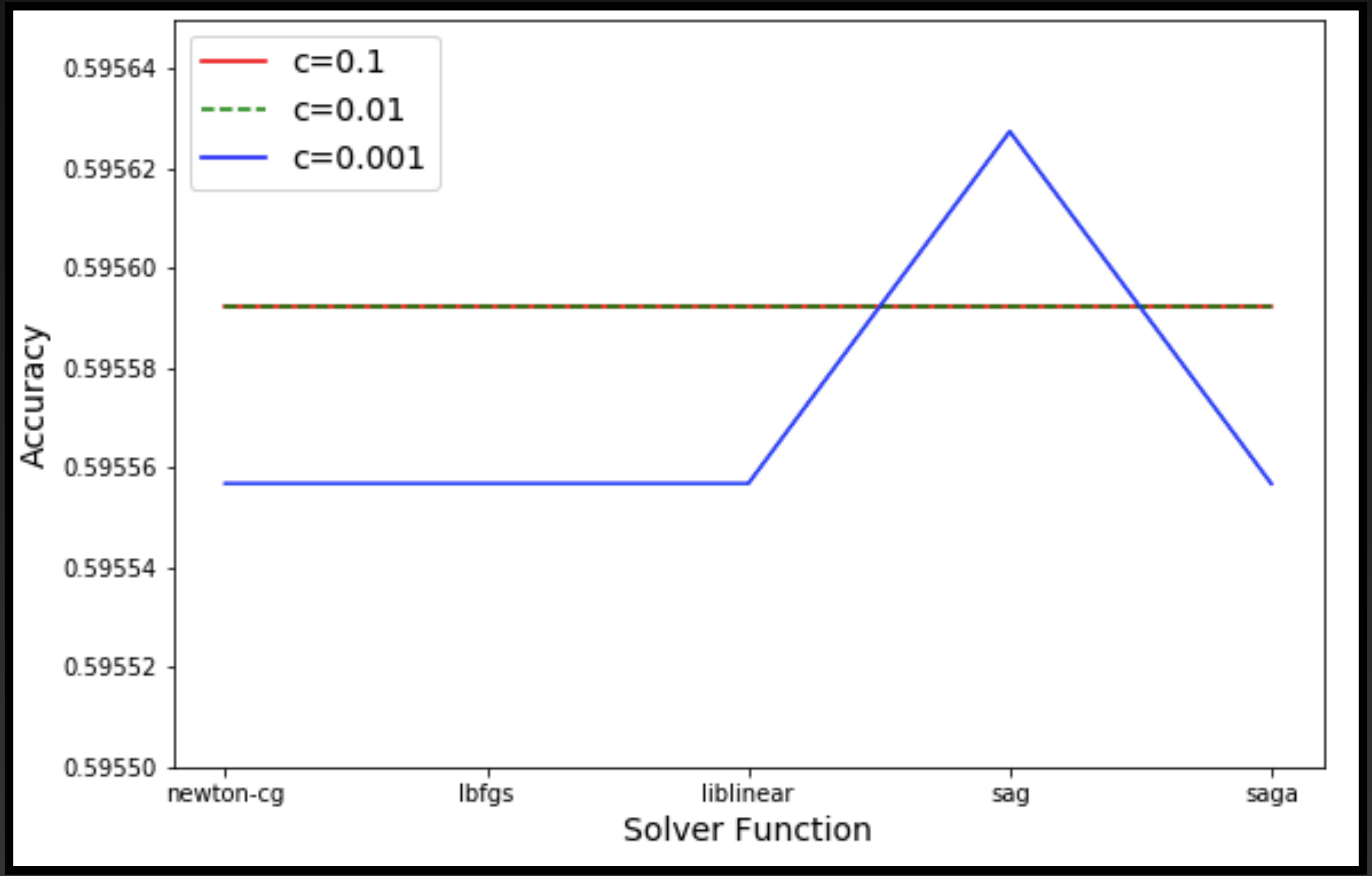
- Varied value of max depth from 2 to 6.
- Used Jaccard Index to determine the best accuracy model.
- Trained the model (d = 5) with train data.
- Evaluated the model using test data.



MODEL BUILDING

Logistic Regression

- Checked the model accuracy for c = 0.1, 0.01, 0.001.
- Varied the solver function for each c.
- Used Jaccard Index to determine the best accuracy model.
- Trained the model (c = 0.001 and sag function) with train data.
- Evaluated the model using test data.



MODEL EVALUATION

Algorithm	Jaccard	F1-score	Log Loss
KNN	0.571292	0.557127	NA
Decision Tree	0.597605	0.591626	NA
Logistic Regression	0.595627	0.588911	0.668231

RESULTS

- Decision tree with max depth = 5 gives the best accuracy model.
- The Jaccard index for the above model = 0.59.

This model can be deployed to predict the severity of a road accident.

THANK YOU..!!!