Business Problem:

Car accidents lead to fatalities and economic losses. Thus, preventing loss of life and property is a topic of concern.

The Seattle Department of Transportation kept the detailed records of car accidents. The big question asked is whether the government can deploy a system that can alert drivers, health system, and police to remind them to practice caution and alertness in case of an incident. To development this model, several key questions we may ask:

- Is the car accident severity associated with objective factors such as weather condition, light condition, certain days or weekend, etc?
- Is the car accident severity associated with transportation infrastructure such as pedestrian side way, road type, cross street setting?
- Is the car accident severity associated with subjective factors related to driver, such as driver's inattention, influence by drugs or alcohol?
- Is the car accident severity happened in certain location?

If we could build up such model to predict the possibility of car accident and how severe it could be, given the key factors related to car accidents. The government can alert people to drive more carefully or even change their travel if possible based on the objective and/or subjective factors involved. Also, the government can take proactive actions to improve the observed the key factors and further reduce the number of car accidents.

Understand the Data:

Seattle Department of Transportation collects all the collision data and records by traffic records. Data includes all the types of collisions like angles, sideswipe, parked cars. These collisions will be displayed at the intersection or mid-block of a segment. The timeframe of data is from 2004 to the present. The data is located in: https://s3.us.cloud-object-storage.appdomain.cloud/cf-courses-data/CognitiveClass/DP0701EN/version-2/Data-Collisions.csv

At a glance of the data:

The total data size is (194673,38). And each attribute type is as blow:

```
SEVERITYCODE        int64
X                 float64
Y                 float64
OBJECTID            int64
INCKEY              int64
COLDETKEY           int64
REPORTNO           object
STATUS             object
ADDRTYPE           object
INTKEY            float64
LOCATION           object
EXCEPTRSNCODE      object
EXCEPTRSNDESC      object
SEVERITYCODE.1      int64
SEVERITYDESC       object
COLLISIONTYPE      object
PERSONCOUNT         int64
PEDCOUNT            int64
PEDCYLCOUNT         int64
VEHCOUNT            int64
INCDATE            object
INCDTTM            object
JUNCTIONTYPE       object
SDOT_COLCODE        int64
SDOT_COLDESC       object
INATTENTIONIND     object
UNDERINFL          object
WEATHER            object
ROADCOND           object
LIGHTCOND          object
PEDROWNOTGRNT      object
SDOTCOLNUM        float64
SPEEDING           object
ST_COLCODE         object
ST_COLDESC         object
SEGLANEKEY          int64
CROSSWALKKEY        int64
HITPARKEDCAR       object
dtype: object
```

From the total 38 attributes:
SEVERITYCODE will be the dependent variable, it's categorical data with value:

0: Little to no Probability (Clear Conditions)

1: Very Low Probability – Chance or Property Damage

2: Low Probability – Chance of Injury

2b: Mild Probability – Chance of Serious Injury

3: High Probability – Chance of Fatality Following is a table of all the attributes along with their data types, variable length and a description for understanding.

From source data, it's distributed as:

All the total data 194673 are all categorized into 1 or 2 severity categories:

	SEVERITYCODE
1	136485
2	58188

As SEVERITYCODE is the predictive variable, and compensate the SEVERITYCODE data will skew the data and model will be inaccuracy. Thus, the model will go with Binary Logistic Regression and/or Decision Tree

Other attribute will be potential Independent variable (also refer to ‘features’ later):

- X, Y is the Longitude and Latitude factor, it will be used for map plotting
- OBJECTID, INCKEY, COLDETKEY, INTKEY and REPORTNO. are the unique identifiers, won't help in the model and will be dropped from data set
- EXCEPTRSNCODE, EXCEPTRSNDESC, STATUS won’t be contributing to our dataset. Will be dropped.
- LOCATION data maybe help us in populating the maps and getting the count in a particular area but the X latitudes and Y longitudes and already in place to serve that purpose.
- SEVERITYCODE.1 and SEVERITYDESC are duplicated with SEVERITYCODE and won't take into the model.
- INCDATE – Incident Date and INCDTTM – Incident Timestamp, INCDATE will be maintained and transfer to certain 'Day of Week' or 'WeekEnd' to judge whether the car accidents related to certain day of week or weekend. The timestamp column doesn’t have consistent values. in this exercise, will not take it as critical features (further study will do regarding question: whether car accident related to certain time of day).
- SDOT_COLCODE and SDOT_COLDESC are redundant, ST_COLCODE, ST_COLDESC, are the repeated features and information only, which shouldn’t be considered in further analysis
- SDOT_COLNUM doesn't have linearrelationship with SEVERITYCODE, drop from the dataset
- COLLISIONTYPE has some missing values. Theoretically, those can be filled by mapping the SDOT_COLDESC values, SDOT_COLDESC involves the collision description. However, by examining the data and found the inconsistent usage of SDOT_COLDESC. And consider the total missing data is 4904 vs total 194673. The none filled value will be dropped
- Features like JUNCTIONTYPE, WEATHER, ROADCOND, LIGHTCOND contain null values, it would be mapped to 'unknown'
- Features like INATTENTIONIND, UNDERINFL, SPEEDING, PEDROWNOTGRNT, HITPARKEDCAR are variables with binary values and has values input for only one class. Thus, we can induce the either value to account for all the blank cells. However, considering the # of UNDERINFL is small, will map its Y value to "INATTENTIONIND" and drop it (with normal philosophy assumption the Under DRUG/ALCOHOL will influence the "INATTENTIONIND" to Yes)
- Features like SEGLANEKEY, CROSSWALKKEY, PEDCOUNT, VEHCOUNT, PEDCYLCOUNT should be examined and normalized, if they aren’t correlated to the target variable then they should be excluded, it can be conceived as noise.

Analytical Techniques

For this project, Github repository (https://github.com/songttony/Coursera_Capstone) is used to commit changes and updation of the code.

IBM Cloud Jupyter Notebook is used for coding and to preprocess data and build Machine Learning models. Python packages included for coding are including:

- Pandas, NumPy and Sklearn for data processing, Standard scaler for data normaliztion, etc..
- Scipy for statistical analysis such as Pearson correlation analysis, Chi-Square analysis, etc.
- Matplotlib, pyplot, seaborn for visulizations, Classification reports as an evaluation metric.

In this car accident severity modeling, the input vectors are the features of the accident, such as driver behavior, road infrastructure and environment characteristics such as weather, light, etc, while the output vector is the corresponding class of accident severity. And the model used will be Logistic Regression/Tree Decision

Methodologies - Exploratory Data Analysis

Data Preparation:

Before exploring the data, notice that there are several issues in the original dataset, such as missing data, data type inconsistent, no useful data, etc.

```
#missing data?
df.replace("?", np.nan, inplace = True)
missing_data = df.isnull()
print("data missing",missing_data.head())
print("data missing size",missing_data.shape)
for column in missing_data.columns.values.tolist():
    print(column)
    print (missing_data[column].value_counts())
    print("")
```

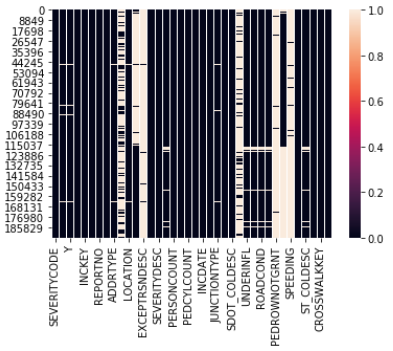
```
data missing    SEVERITYCODE      X      Y  OBJECTID  INCKEY  COLDETKEY  REPORTNO  STATUS  \
0          False  False  False     False  False      False      False     False   False
1          False  False  False     False  False      False      False     False   False
2          False  False  False     False  False      False      False     False   False
3          False  False  False     False  False      False      False     False   False
4          False  False  False     False  False      False      False     False   False

   ADDRTYPE  INTKEY  ...  ROADCOND  LIGHTCOND  PEDROWNOTGRNT  SDOTCOLNUM  \
0     False   False  ...     False      False           True        True
1     False    True  ...     False      False           True       False
2     False    True  ...     False      False           True       False
3     False    True  ...     False      False           True        True
4     False   False  ...     False      False           True       False

   SPEEDING  ST_COLCODE  ST_COLDESC  SEGLANEKEY  CROSSWALKKEY  HITPARKEDCAR
0      True       False       False       False         False         False
1      True       False       False       False         False         False
2      True       False       False       False         False         False
3      True       False       False       False         False         False
4      True       False       False       False         False         False
```

With that in mind, first, filling the missing data, convert the date type and drop the NaN data value.

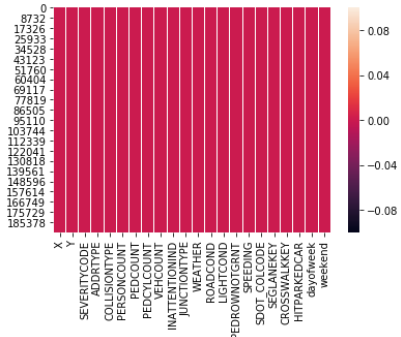
```
#Filling missing data for the categorical data
df["PEDROWNOTGRNT"].replace(np.nan, 'N', inplace=True)
df["UNDERINFL"].replace(np.nan, 'N', inplace=True)
df["SPEEDING"].replace(np.nan, 'N', inplace=True)
df["UNDERINFL"].replace('0', 'N', inplace=True)

#Inattention missing data: with Underinfle factor, assuming Underinfle = Y, the missing inattention shall be "Y". Others are "N"
df["INATTENTIONIND"].replace(np.nan, 'N', inplace=True)
df.loc[df["UNDERINFL"]=='Y','INATTENTIONIND'] = 'Y'
#filling weathercond,roadcond and lightcond with 'Unknown' considering the amount is not that significant
#convert to low string
df['ROADCOND'] = df['ROADCOND'].str.lower()
df['WEATHER'] = df['WEATHER'].str.lower()
df['LIGHTCOND'] = df['LIGHTCOND'].str.lower()
df['ADDRTYPE'] = df['ADDRTYPE'].str.lower()
df['JUNCTIONTYPE'] = df['JUNCTIONTYPE'].str.lower()
df["ROADCOND"].replace(np.nan, 'unknown', inplace=True)
df["WEATHER"].replace(np.nan, 'unknown', inplace=True)
df["LIGHTCOND"].replace(np.nan, 'unknown', inplace=True)
df["JUNCTIONTYPE"].replace(np.nan, 'unknown', inplace=True)
#check whether it is related to particular day of week or weekend
df['INCDATE'] = pd.to_datetime(df['INCDATE'])
df['dayofweek'] = df['INCDATE'].dt.dayofweek
df['weekend'] = df['dayofweek'].apply(lambda x: 1 if (x>3)  else 0)
df.head()
```

	SEVERITYCODE	X	Y	OBJECTID	INCKEY	COLDETKEY	REPORTNO	STATUS	ADDRTYPE	INTKEY	...	PEDROWNOTGRNT	SDOTCOLNUM	SPEEDING	ST_COLCODE	ST_COLDESC	SEGLANEKEY	CROSSWALKKEY	HITPARKED
0	2	-122.323148	47.703140	1	1307	1307	3502005	Matched	intersection	37475.0	...	N	NaN	N	10	Entering at angle	0	0	
1	1	-122.347294	47.647172	2	52200	52200	2607959	Matched	block	NaN	...	N	6354039.0	N	11	From same direction - both going straight - bo...	0	0	
2	1	-122.334540	47.607871	3	26700	26700	1482393	Matched	block	NaN	...	N	4323031.0	N	32	One parked--one moving	0	0	
3	1	-122.334803	47.604803	4	1144	1144	3503937	Matched	block	NaN	...	N	NaN	N	23	From same direction - all others	0	0	
4	2	-122.306426	47.545739	5	17700	17700	1807429	Matched	intersection	34387.0	...	N	4028032.0	N	10	Entering at	0	0	

After this is done, the missing data value is compared as below two pictures:



Before VS. After



And then performed the statistical analysis to understand the correlation between different features and predictive variable [SEVERITYCODE]. For example, Chi-Square used to test the categorical features, e.g ROADCOND correlation with SEVERITYCODE

```
sub_df2 = df.loc[:,['ROADCOND','SEVERITYCODE']]

chi_table = pd.crosstab(index=sub_df2['ROADCOND'],columns=sub_df2['SEVERITYCODE'])
tstatics,p,dof,ex = chi2_contingency(chi_table)
print ('Chi-Square test statistic ROADCOND vs SEVERITYCODEis',tstatics,'with P-value:',p)
```

```
Chi-Square test statistic ROADCOND vs SEVERITYCODEis 4856.908569911719 with P-value: 0.0
```

The Chi-Square analysis demontrated that SEVEIRTY CODE is associated with ROADCOND

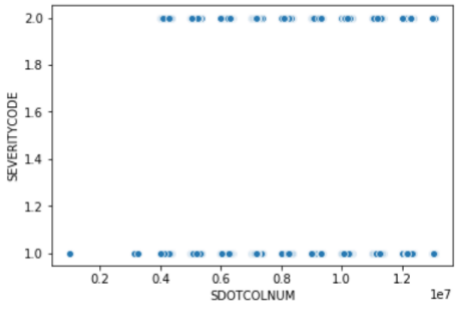
and Pearsons Correlation analysis for continuous features. E.g SDOTCOLNUM, with Pearsons output P-value, the NULL hypothesis was rejected and there is not linear relationship between SEVERITYCODE and SDOTCOLNUM. Thus, SDOTCOLNUM was removed from future analysis.

```
pearson_coef, p_value = stats.pearsonr(df['SEVERITYCODE'], df['SDOTCOLNUM'])
print ('The coefficiency for SDOTCOLNUM is',pearson_coef, 'with P-value:',p_value)
```

```
The coefficiency for SDOTCOLNUM is nan with P-value: 1.0
```

```
sns.scatterplot(x='SDOTCOLNUM',y='SEVERITYCODE',data=df)
```

```
]: <matplotlib.axes._subplots.AxesSubplot at 0x7f1592f46748>
```



After that, except the necessary features dataset from the raw data. Since most machine learning algorithms require the number data as input. We will perform the

encode the categorical features by using sklearn.proprocessing.LabelEncoder() method to encode the data.

```
#Encode the categorical value
from sklearn import preprocessing
from sklearn.preprocessing import LabelEncoder
dataset = Features
cat_encode = LabelEncoder()
dataset['ADDRTYPE'] = cat_encode.fit_transform(dataset['ADDRTYPE'])
dataset['INATTENTIONIND'] = cat_encode.fit_transform(dataset['INATTENTIONIND'])
dataset['WEATHER'] = cat_encode.fit_transform(dataset['WEATHER'])
dataset['ROADCOND'] = cat_encode.fit_transform(dataset['ROADCOND'])
dataset['LIGHTCOND'] = cat_encode.fit_transform(dataset['LIGHTCOND'])
dataset['PEDROWNOTGRNT'] = cat_encode.fit_transform(dataset['PEDROWNOTGRNT'])
dataset['SPEEDING'] = cat_encode.fit_transform(dataset['SPEEDING'])
dataset['PEDROWNOTGRNT'] = cat_encode.fit_transform(dataset['PEDROWNOTGRNT'])
dataset['JUNCTIONTYPE'] = cat_encode.fit_transform(dataset['JUNCTIONTYPE'])
dataset['COLLISIONTYPE'] = cat_encode.fit_transform(dataset['COLLISIONTYPE'])
dataset['HITPARKEDCAR'] = cat_encode.fit_transform(dataset['HITPARKEDCAR'])
```

```
dataset.head()
```

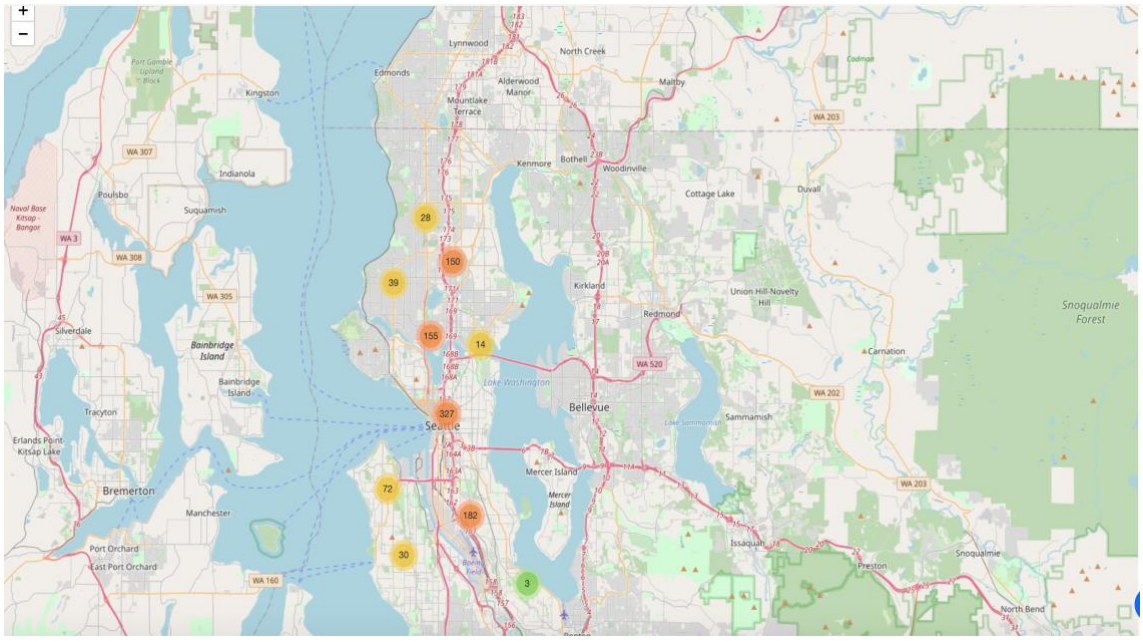
	X	Y	SEVERITYCODE	ADDRTYPE	COLLISIONTYPE	PERSONCOUNT	PEDCOUNT	PEDCYLCOUNT	VEHCOUNT	INATTENTIONIND	...	ROADCOND	LIGHTCOND	PEDROWNOTGRNT	SPEEDING	SDOT_COLCODE	SEGLANEKEY
0	-122.323148	47.703140	2	1	0	2	0	0	2	0	...	8	5	0	0	11	0
1	-122.347294	47.647172	1	0	9	2	0	0	2	0	...	8	2	0	0	16	0
2	-122.334540	47.607871	1	0	5	4	0	0	3	0	...	0	5	0	0	14	0
3	-122.334803	47.604803	1	0	4	3	0	0	3	0	...	0	5	0	0	11	0
4	-122.306426	47.545739	2	1	0	2	0	0	2	0	...	8	5	0	0	11	0

5 rows × 22 columns

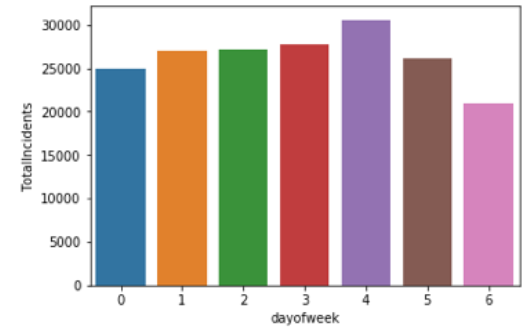
Now, data set is ready for next step.

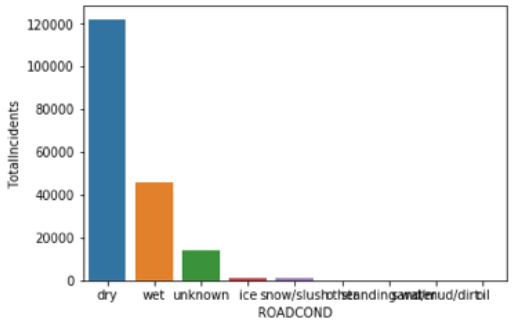
Data Visualization:

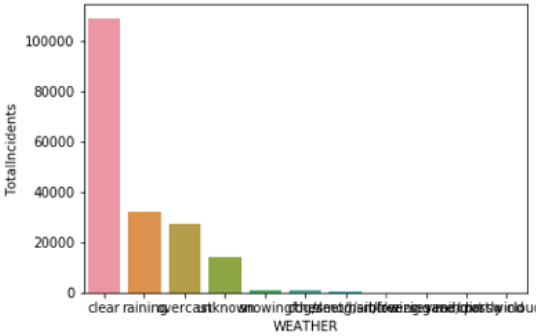
From the incidents data and its Longitude (X) and Latitude (Y) information, the Seattle Open Street map will indicate the most hot area with highest number of car accidents, which will alert drivers to be more careful in these areas. And according, the police station, hospitals could also arrange more manpower and facilities to fast respond to the accidents (for illustration purpose, only the first 1000 data points were plotted). From the map, we can observe the most car accidents happened along the mainroads.

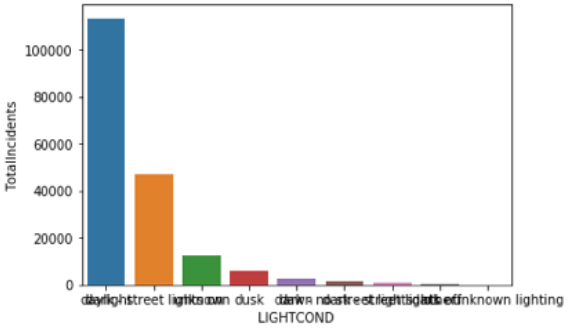


Further checking the relationship between Features: Day of week, Road Condition, Weather Condition, Light Condition and the number of car accidents. We can observe there are certain factors with significant number of car accidents associated.

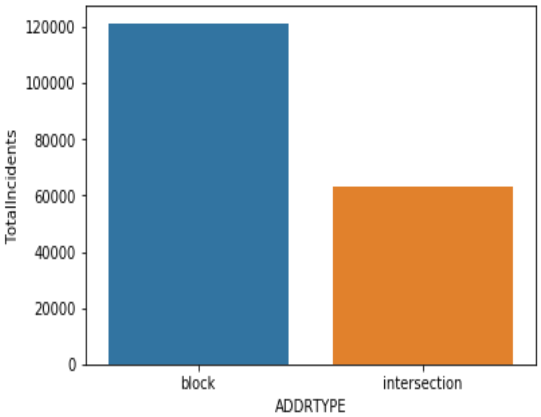


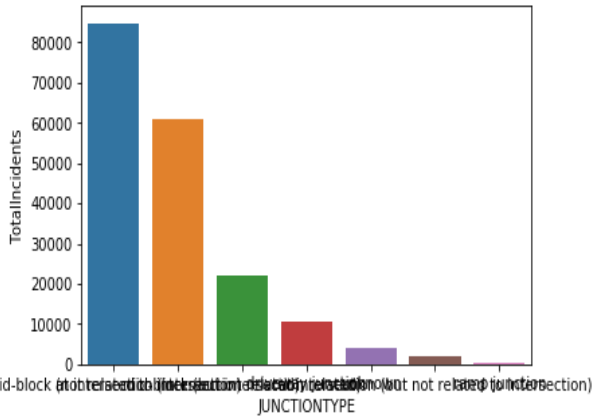






Similarly, we can also plot the bar chart of Road type and Junction Type, “Block” – ‘mid-block’, intersection occupy significant number of car accidents.





Statistic correlation can be observed here:

	X	Y	SEVERITYCODE	ADDRTYPE	COLLISIONTYPE	PERSONCOUNT	PEDCOUNT	PEDCYLCOUNT	VEHCOUNT	INATTENTIONIND	...	ROADCOND	LIGHTCOND	PEDROWNOTGRNT	SPEEDING	SDOT_COLCODE
X	1.000000	-0.160095	0.010375	0.005393	0.010696	0.012787	0.011346	-0.001801	-0.015710	-0.022033	...	0.000267	0.010026	0.008413	-0.008364	0.011004
Y	-0.160095	1.000000	0.018337	0.034231	-0.038915	-0.014000	0.010344	0.026728	0.020630	0.006281	...	0.008060	0.024464	0.022287	-0.026911	-0.019808
SEVERITYCODE	0.010375	0.018337	1.000000	0.200601	-0.129741	0.127633	0.248506	0.216213	-0.085068	0.051586	...	-0.046302	-0.057011	0.208597	0.037518	0.182845
ADDRTYPE	0.005393	0.034231	0.200601	1.000000	-0.481823	0.062622	0.145904	0.084045	-0.095974	-0.084555	...	-0.026710	-0.051536	0.157336	-0.056752	-0.035102
COLLISIONTYPE	0.010696	-0.038915	-0.129741	-0.481823	1.000000	0.012533	0.094323	-0.213623	0.103101	0.112976	...	-0.001528	0.030481	-0.020357	-0.003201	0.002707
PERSONCOUNT	0.012787	-0.014000	0.127633	0.062622	0.012533	1.000000	-0.026978	-0.042390	0.395466	0.071808	...	-0.028803	-0.039712	-0.030526	-0.001623	-0.142715
PEDCOUNT	0.011346	0.010344	0.248506	0.145904	0.094323	-0.026978	1.000000	-0.018241	-0.321469	-0.002146	...	0.007042	-0.039898	0.495414	-0.033370	0.270182
PEDCYLCOUNT	-0.001801	0.026728	0.216213	0.084045	-0.213623	-0.042390	-0.018241	1.000000	-0.310774	-0.001879	...	-0.049668	0.012931	0.322998	-0.020694	0.397357
VEHCOUNT	-0.015710	0.020630	-0.085068	-0.095974	0.103101	0.395466	-0.321469	-0.310774	1.000000	0.042196	...	-0.013608	0.030351	-0.280326	-0.040702	-0.468722
INATTENTIONIND	-0.022033	0.006281	0.051586	-0.084555	0.112976	0.071808	-0.002146	-0.001879	0.042196	1.000000	...	-0.052388	-0.057893	-0.032629	-0.026465	0.052844
JUNCTIONTYPE	-0.014623	-0.039897	-0.218741	-0.896017	0.461905	-0.083400	-0.129494	-0.088685	0.085875	0.064088	...	0.056101	0.081341	-0.152030	0.047870	-0.030642
WEATHER	-0.010787	0.022552	-0.105838	-0.086801	0.028128	-0.061658	-0.010212	-0.053547	-0.008399	-0.085188	...	0.747192	0.204641	-0.015177	0.034565	-0.104712
ROADCOND	0.000267	0.008060	-0.046302	-0.026710	-0.001528	-0.028803	0.007042	-0.049668	-0.013608	-0.052388	...	1.000000	0.021926	-0.000388	0.082982	-0.042166
LIGHTCOND	0.010026	0.024464	-0.057011	-0.051536	0.030481	-0.039712	-0.039898	0.012931	0.030351	-0.057893	...	0.021926	1.000000	-0.014726	-0.096371	-0.149808
PEDROWNOTGRNT	0.008413	0.022287	0.208597	0.157336	-0.020357	-0.030526	0.495414	0.322998	-0.280326	-0.032629	...	-0.000388	-0.014726	1.000000	-0.028913	0.247309
SPEEDING	-0.008364	-0.026911	0.037518	-0.056752	-0.003201	-0.001623	-0.033370	-0.020694	-0.040702	-0.026465	...	0.082982	-0.096371	-0.028913	1.000000	0.137527
SDOT_COLCODE	0.011004	-0.019808	0.182845	-0.035102	0.002707	-0.142715	0.270182	0.397357	-0.468722	0.052844	...	-0.042166	-0.149808	0.247309	0.137527	1.000000
SEGLANEKEY	-0.001803	0.004545	0.104947	0.037912	-0.097611	-0.022650	0.001427	0.456772	-0.147298	-0.002795	...	-0.022206	0.004588	0.152594	-0.006573	0.212427
CROSSWALKKEY	0.014233	0.009637	0.176489	0.178108	0.033983	-0.034727	0.571162	0.109711	-0.241565	-0.003857	...	0.009906	-0.022734	0.451361	-0.025631	0.194226
HITPARKEDCAR	-0.016419	-0.012385	-0.102051	-0.130499	0.036961	-0.052890	-0.035661	-0.030453	0.069331	0.027361	...	0.019863	0.034722	-0.031184	-0.026578	-0.096702
dayofweek	-0.008652	-0.005819	-0.017183	-0.011869	-0.016799	0.058679	-0.020704	-0.026050	0.005971	0.003659	...	0.019355	-0.093844	-0.029712	0.030606	0.009995
weekend	-0.008449	-0.005699	-0.018251	-0.010251	-0.014111	0.057849	-0.017856	-0.028079	0.004667	0.001994	...	0.015659	-0.087236	-0.028076	0.028479	0.008453

Methodologies – Build & Evaluate the Model

- **Train and Test data split**

  Data will be split with 70% used for train data and 30% used for verifying the model purpose. Before doing that, data will be standardized.

```
#normalize the data
X = preprocessing.StandardScaler().fit(X).transform(X)
X[0:5]
```

The final data set size is as the following:

```
Train set: (129207, 19) (129207,)
Test set: (55375, 19) (55375,)
```

- **Build the model**

  "LogisticRegression" and "DecisionTreeClassifier" are used to build the model

- **Evaluate the two model:**

  The logistic regression model's evaluation results:

```
the Logistic Loss value is:  0.5213520892767424
the Jaccard Score is:  0.7507178329571106
the F1 Score is:  0.8439555967534083
```

  The Decision Tree model's evaluation results:

```
Avg F1-score: 0.7168
Jaccard score: 0.7581
```

- **Results:**

  By comparing the two models, logistic regression will be the better choice: with around 84% accuracy ratio, actually, it is not a bad model.

Discussion:

In this study, the relationship between Severity code and different attributes in the car accidents' data were analyzed. And with statistic analysis (Chi-square and Pearsons), I have identified light conditions, road conditions, weather, inattention and junction types, day of week, weekend, pedrownotgrnt, collisiontype, sdot_colcode, hitparkcar are the significant subjective and objective factors associated with a car collision. Also built both regression models and classification models (decision tree) to predict the severity of car accidents. With the current trained data, Logistic regression model was selected. This model can be beneficial to SDOT, police, and drivers in several ways:

- Help SDOT to take improvement actions, like:
  - installing more traffic signals on particular junction types in block and intersections, or
  - imposing speed limits in more collision causing areas, or
  - Build more sidewalk, or
  - More attention to pedestrians and parked cars, etc.
- Help drivers be more alert while choosing roads, driving more carefully to avoid more collision causing junction types and plan the departure time with proper day of week.

Conclusion

We can conclude that, with current available data, the Logistic Regression is the best model in this scenario. An interesting point to note here is the data set doesn't contain any severity code that are more than 2 (and since it is the predicted variable, we couldn't compensate the it with 'cooked' data!). If there are, I would suspect the Logistic Regression model would not take this significant fit. This needs further study after more data is available.

GitHub link -
https://github.com/songttony/Coursera_Capstone/blob/master/IBM%20DataScience_Capstone.ipynb