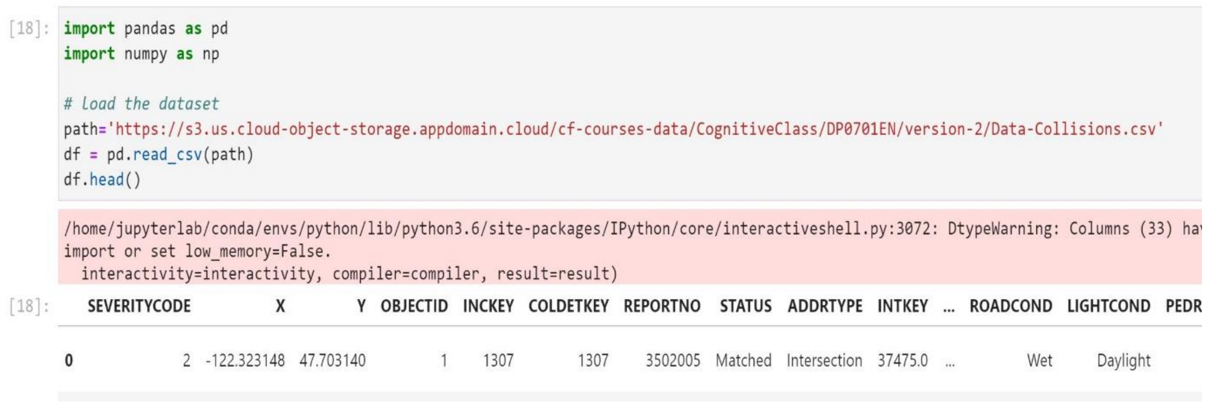
CAR ACCIDENT SEVERITY REPORT

BUSINESS UNDERSTANDING

To reduce the frequency of car collisions at a location, current weather, road, and visibility conditions should be taken into account and an algorithm should be developed to estimate the material and spiritual seriousness of the accident. The main purpose of this study is to prevent accidents when conditions are bad, so this model helps protect drivers from the risks they will encounter. Another possible target audience for this study is as follows. Local police, health institutes, insurance companies, etc. They can use this model well and take action to know when they will be ready to receive bad news about a particular road. In many cases, carelessness while driving, using drugs and alcohol, or driving too fast are some of the main causes of accidents that can be prevented by applying stronger regulations. Besides the above reasons, weather, visibility, or road conditions are the main uncontrollable factors that can be avoided by revealing hidden patterns in the data and reporting a warning to the local government, police, and road drivers.



```
[18]: import pandas as pd
      import numpy as np

      # Load the dataset
      path='https://s3.us.cloud-object-storage.appdomain.cloud/cf-courses-data/CognitiveClass/DP0701EN/version-2/Data-Collisions.csv'
      df = pd.read_csv(path)
      df.head()
```

```
/home/jupyterlab/conda/envs/python/lib/python3.6/site-packages/IPython/core/interactiveshell.py:3072: DtypeWarning: Columns (33) hav
import or set low_memory=False.
  interactivity=interactivity, compiler=compiler, result=result)
```

```
[18]:   SEVERITYCODE           X          Y  OBJECTID  INCKEY  COLDETKEY  REPORTNO  STATUS   ADDRTYPE      INTKEY  ...  ROADCOND  LIGHTCOND  PEDR
      0            2  -122.323148  47.703140         1    1307       1307   3502005  Matched  Intersection  37475.0  ...       Wet   Daylight
```

DATA UNDERSTANDING

The data was collected by the Seattle Police Department and Accident Traffic Records Department from 2004 to present. 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

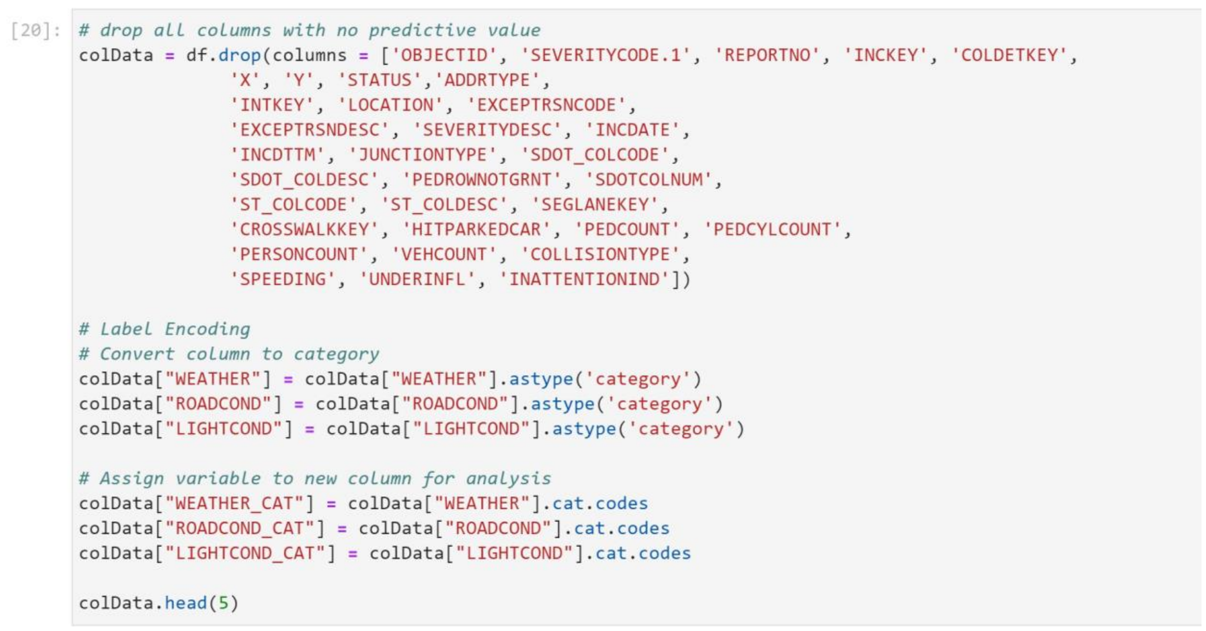
3: Mild Probability - Chance of Serious Injury

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:

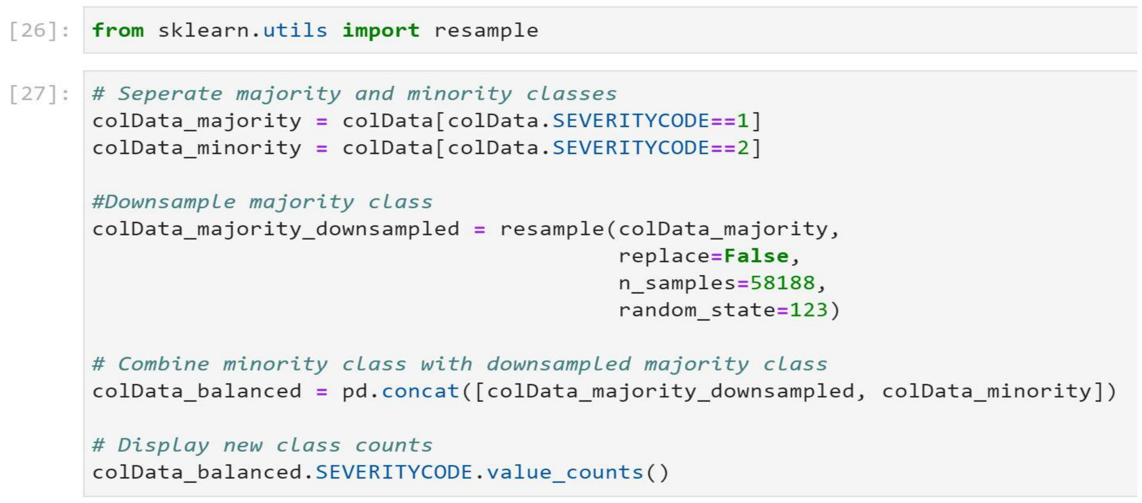
We have to select the most important features to weigh the severity of accidents in Seattle. Among all the features, the following features have the most influence in the accuracy of the predictions: The ‘WEATHER’, ‘ROADCOND’ and ‘LIGHTCOND’ attributes.



```
[20]: # drop all columns with no predictive value
      colData = df.drop(columns = ['OBJECTID', 'SEVERITYCODE.1', 'REPORTNO', 'INCKEY', 'COLDETKEY',
                    'X', 'Y', 'STATUS','ADDRTYPE',
                    'INTKEY', 'LOCATION', 'EXCEPTRSNCODE',
                    'EXCEPTRSNDESC', 'SEVERITYDESC', 'INCDATE',
                    'INCDTTM', 'JUNCTIONTYPE', 'SDOT_COLCODE',
                    'SDOT_COLDESC', 'PEDROWNOTGRNT', 'SDOTCOLNUM',
                    'ST_COLCODE', 'ST_COLDESC', 'SEGLANEKEY',
                    'CROSSWALKKEY', 'HITPARKEDCAR', 'PEDCOUNT', 'PEDCYLCOUNT',
                    'PERSONCOUNT', 'VEHCOUNT', 'COLLISIONTYPE',
                    'SPEEDING', 'UNDERINFL', 'INATTENTIONIND'])

      # Label Encoding
      # Convert column to category
      colData["WEATHER"] = colData["WEATHER"].astype('category')
      colData["ROADCOND"] = colData["ROADCOND"].astype('category')
      colData["LIGHTCOND"] = colData["LIGHTCOND"].astype('category')

      # Assign variable to new column for analysis
      colData["WEATHER_CAT"] = colData["WEATHER"].cat.codes
      colData["ROADCOND_CAT"] = colData["ROADCOND"].cat.codes
      colData["LIGHTCOND_CAT"] = colData["LIGHTCOND"].cat.codes

      colData.head(5)
```

Preprocessing

In this section, we have count the missing values of the attribute columns that we are using to weigh the severity of the collisions we’re going to study. After that we were presented with another imbalanced data. Class 1 was nearly three times larger than class 2. We downsampled to match the minority class exactly with 58188 values each with sklearn's resample tool.



```
[26]: from sklearn.utils import resample

[27]: # Seperate majority and minority classes
      colData_majority = colData[colData.SEVERITYCODE==1]
      colData_minority = colData[colData.SEVERITYCODE==2]

      #Downsample majority class
      colData_majority_downsampled = resample(colData_majority,
                                              replace=False,
                                              n_samples=58188,
                                              random_state=123)

      # Combine minority class with downsampled majority class
      colData_balanced = pd.concat([colData_majority_downsampled, colData_minority])

      # Display new class counts
      colData_balanced.SEVERITYCODE.value_counts()
```

METHODOLOGY

I have employed three of the machine learning models for classification that have been studied in the last course of this specialization:

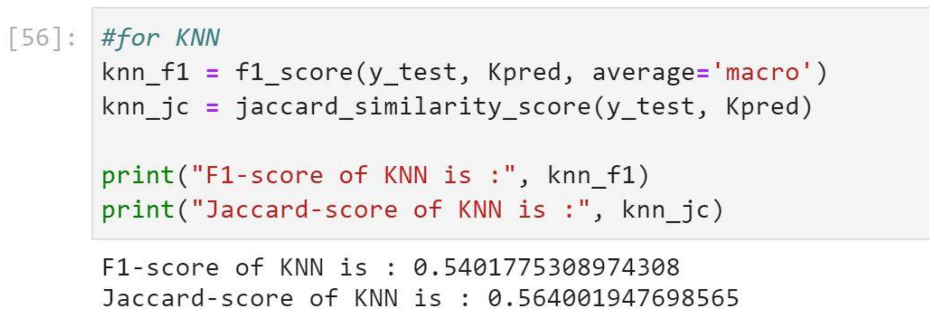
1. K-Nearest Neighbors (KNN)

2. Decision Tree

3. Linear Regression

K-Nearest Neighbors

KNN will help us predict the severity code of an outcome by finding the most similar to data point within k distance.

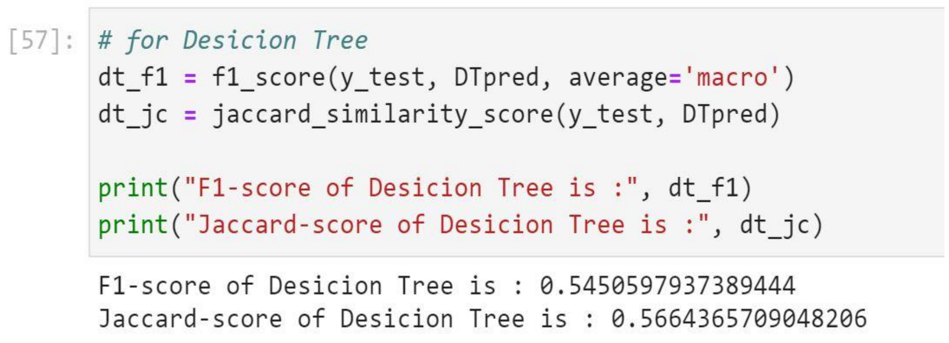


```
[56]: #for KNN
      knn_f1 = f1_score(y_test, Kpred, average='macro')
      knn_jc = jaccard_similarity_score(y_test, Kpred)

      print("F1-score of KNN is :", knn_f1)
      print("Jaccard-score of KNN is :", knn_jc)
```

```
F1-score of KNN is : 0.5401775308974308
Jaccard-score of KNN is : 0.564001947698565
```

Decision Tree

A decision tree model gives us a layout of all possible outcomes so we can fully analyze the consequences of a decision. It context, the decision tree observes all possible outcomes of different weather conditions.

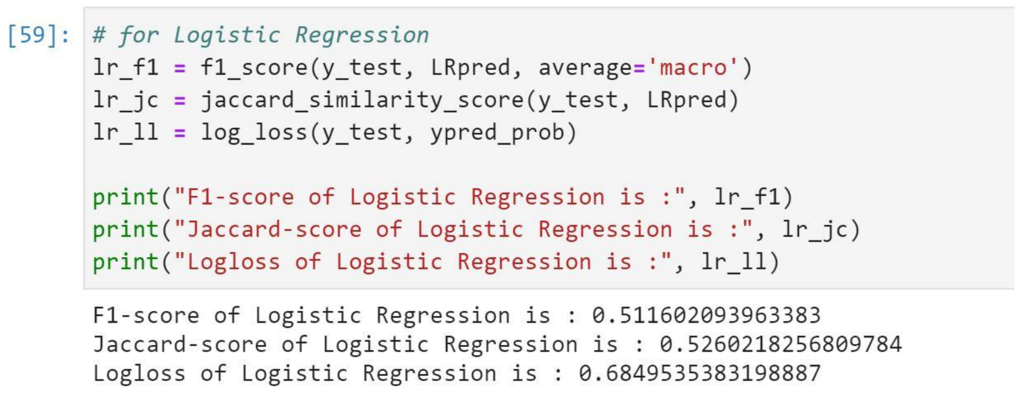


```
[57]: # for Desicion Tree
      dt_f1 = f1_score(y_test, DTpred, average='macro')
      dt_jc = jaccard_similarity_score(y_test, DTpred)

      print("F1-score of Desicion Tree is :", dt_f1)
      print("Jaccard-score of Desicion Tree is :", dt_jc)
```

```
F1-score of Desicion Tree is : 0.5450597937389444
Jaccard-score of Desicion Tree is : 0.5664365709048206
```

Linear Regression

Because our dataset only provides us with two severity code outcomes, our model will only predict one of those two classes. This makes our data binary, which is perfect to use with logistic regression.

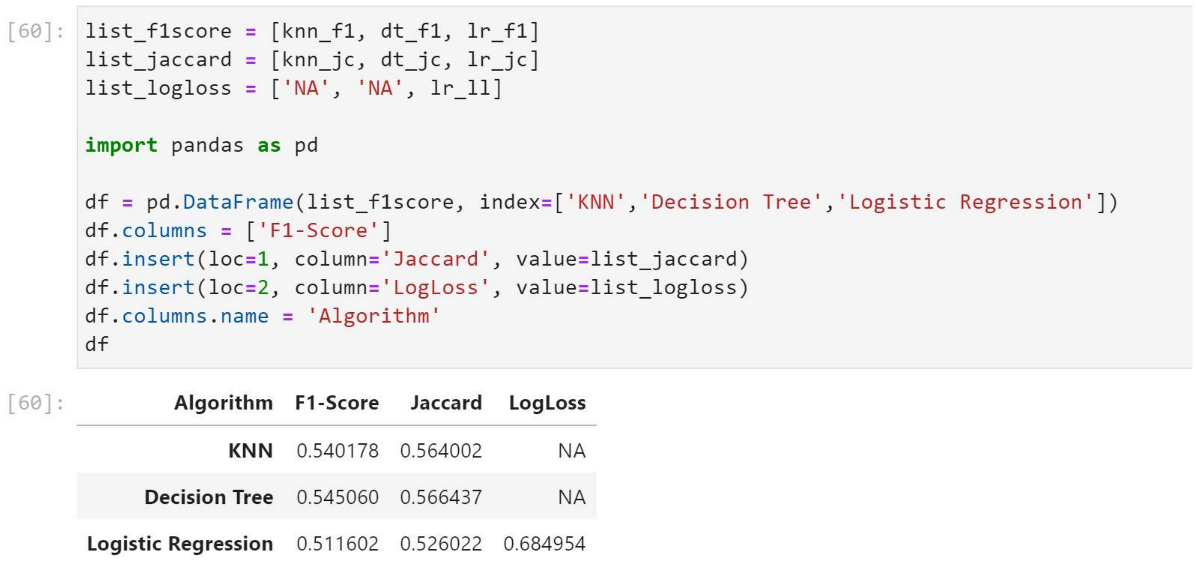


```
[59]: # for Logistic Regression
      lr_f1 = f1_score(y_test, LRpred, average='macro')
      lr_jc = jaccard_similarity_score(y_test, LRpred)
      lr_ll = log_loss(y_test, ypred_prob)

      print("F1-score of Logistic Regression is :", lr_f1)
      print("Jaccard-score of Logistic Regression is :", lr_jc)
      print("Logloss of Logistic Regression is :", lr_ll)
```

```
F1-score of Logistic Regression is : 0.511602093963383
Jaccard-score of Logistic Regression is : 0.5260218256809784
Logloss of Logistic Regression is : 0.6849535383198887
```

RESULTS & EVALUATION

The final results of the model evaluations are summarized in the following table:



```
[60]: list_f1score = [knn_f1, dt_f1, lr_f1]
      list_jaccard = [knn_jc, dt_jc, lr_jc]
      list_logloss = ['NA', 'NA', lr_ll]

      import pandas as pd

      df = pd.DataFrame(list_f1score, index=['KNN','Decision Tree','Logistic Regression'])
      df.columns = ['F1-Score']
      df.insert(loc=1, column='Jaccard', value=list_jaccard)
      df.insert(loc=2, column='LogLoss', value=list_logloss)
      df.columns.name = 'Algorithm'
      df
```

[60]:

Algorithm	F1-Score	Jaccard	LogLoss
KNN	0.540178	0.564002	NA
Decision Tree	0.545060	0.566437	NA
Logistic Regression	0.511602	0.526022	0.684954

Based on the results table, Decision Tree is the best model to predict car accident severity.

DISCUSSION

Firstly, we had categorical data that was of type 'object'. This is not a data type that we could have fed through an algorithm, so label encoding was used to created new classes that were of type int8; a numerical data type. Secondly, resolving this issue we were presented with another imbalanced data. Class 1 was nearly three times larger than class 2. We downsampled to match the minority class exactly with 58188 values each with sklearn's resample tool. Thirdly, we analyzed and cleaned the data, it was then fed through three ML models; K-Nearest Neighbor, Decision Tree and Logistic Regression. Finally, evaluation metrics used to test the accuracy of our models were jaccard index, f-1 score and logloss for logistic regression. Choosing different k, max depth and hyperamater C values helped to improve our accuracy to be the best possible. We have reached the correct result by modifying the values.

CONCLUSION

As a result, we can conclude that certain weather conditions have some effect on whether the trip will result in property damage (class 1) or injury (class 2). Although the rainy weather driving effect on the probability of injury, certain parts of the city, blind spots, there may be other factors, such as lightning conditions and more. The change in the speed limit or it may be wise to create more awareness of drivers to understand the dangerous driving conditions and improved it appropriate models.