CAR SEVERITY PROJECT
BY Tamalika Maity

Introduction/Business Problem:

- The Seattle government is going to prevent avoidable car accidents by employing methods that alert drivers, health system, and police to remind them to be more careful in critical situations. In most cases, not paying enough attention during driving, abusing drugs and alcohol or driving at very high speed are the main causes of occurring accidents that can be prevented by enacting harsher regulations. Apart from these , weather, visibility, or road conditions are the major uncontrollable factors that can be prevented by revealing hidden patterns in the data and announcing warning to the local government, police and drivers on the targeted roads.

- The target audience of the project is local Seattle government, police, rescue groups, and last but not least, car insurance institutes. The model and its results are going to provide some advice for the target audience to make insightful decisions for reducing the number of accidents and injuries for the city.

- Using the collisions data provided by Coursera for the final capstone course, I wanted to look into the severity of car accidents that are due to bad weather conditions. How many car accidents are caused due to bad weather conditions? This data will bring awareness to people to drive extra cautiously during bad weather!

Data

- 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 null values in some records, the data needs to be preprocessed before any further processing.

Data Preprocessing

- We can see that dataset in the original form is not ready for data analysis. In order to prepare the data, first, we need to drop the non-relevant columns. In addition, most of the features are of object data types that need to be converted into numerical data types. After analyzing the data set, I have decided to focus on only four features, severity, weather conditions, road conditions, and light conditions, among others.

- To get a good understanding of the dataset, I have checked different values in the features. The results show, the target feature is imbalance, so we use a simple statistical technique to balance it.

```
[19]: df["SEVERITYCODE"].value_counts()

[19]: 1     136485
      2      58188
      Name: SEVERITYCODE, dtype: int64
```

As you can see, the number of rows in class 1 is almost three times bigger than the number of rows in class 2. It is possible to solve the issue by down sampling the class 1.

```
21]: from sklearn.utils import resample

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

     df_maj_dsample= resample(df_maj,
                              replace=False,
                              n_samples=58188,
                              random_state=123)

     balanced_df= pd.concat([df_maj_dsample,df_min])

     balanced_df.SEVERITYCODE.value_counts()

25]: 2     58188
     1     58188
     Name: SEVERITYCODE, dtype: int64
```

Methodology

- For implementing the solution, I have used Github as a repository and running Jupyter Notebook to preprocess data and build Machine Learning models. For coding, I have used Python and its popular packages such as Pandas, NumPy and Sklearn.

- Once I have loaded the data into Pandas Dataframe, I used *'dtypes'* attribute to check the feature names and their data types. Then I have selected the most important features to predict the severity of accidents in Seattle. Among all the features, the following features have the most influence in the accuracy of the predictions:

- "WEATHER",

- "ROADCOND",

- "LIGHTCOND"

- Also, as I mentioned earlier, "SEVERITYCODE" is the target variable.
- I have run a value count on road ('ROADCOND') and weather condition ('WEATHER') to get ideas of the different road and weather conditions. I also have run a value count on light condition ('LIGHTCOND'), to see the breakdowns of accidents occurring during the different light conditions. The results can be seen below:

```
[26]: df["WEATHER"].value_counts()
```

```
[26]: 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
```

```
[27]: df["ROADCOND"].value_counts()
```

```
[27]: 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
```

```
[28]: df["LIGHTCOND"].value_counts()
```

```
[28]: 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
```

- **After balancing SEVERITYCODE feature, and standardizing the input feature, the data has been ready for building machine learning models.**
- I have employed three machine learning models: Logistic regression
- Decision Tree random forest After importing necessary packages and splitting preprocessed data into test and train sets, for each machine learning model, I have built and evaluated the model and shown the results as follow:

Using Logistic Regression Model

```python
[67]: from sklearn.linear_model import LogisticRegression
      from sklearn.metrics import plot_confusion_matrix
      from sklearn.model_selection import train_test_split
      from sklearn.metrics import classification_report



      LR = LogisticRegression(max_iter=100000)


      X_train, X_test, y_train, y_test = train_test_split(X,y,test_size=0.20)


      LR.fit(X_train,y_train)
      score = LR.score(X_test, y_test)
      print(score)


      #y_pred = LR.predict(X_test)



      plot_confusion_matrix(LR,X_test,y_test)

      y_pred = LR.predict(X_test)
      print(classification_report(y_test, y_pred))
```
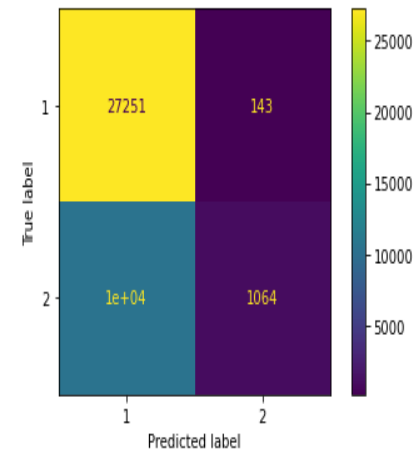
```
0.7272377038654168
            precision    recall  f1-score   support
```

```
0.7272377038654168
              precision    recall  f1-score   support

           1       0.72      0.99      0.84     27394
           2       0.88      0.09      0.17     11541

    accuracy                           0.73     38935
   macro avg       0.80      0.54      0.50     38935
weighted avg       0.77      0.73      0.64     38935
```

Using Decision Tree Classifier

```
[9]:  from sklearn.tree import DecisionTreeClassifier
      from sklearn.model_selection import  cross_val_score

      DT = DecisionTreeClassifier()

      DT.fit(X_train,y_train)
      score_1 = DT.score(X_test, y_test)
      print(score_1)
      from sklearn.metrics import plot_confusion_matrix
      plot_confusion_matrix(DT,X_test,y_test)

      y_pred_1 = DT.predict(X_test)

      from sklearn.metrics import classification_report
      print(classification_report(y_test, y_pred_1))
```
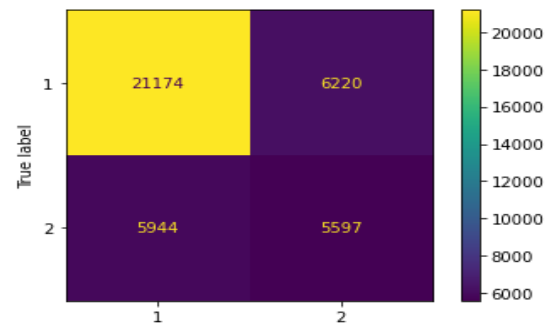
```
0.6875818672145885
              precision    recall  f1-score   support

           1       0.78      0.77      0.78     27394
           2       0.47      0.48      0.48     11541
```

```
0.68758186/2145885
              precision    recall  f1-score   support

           1       0.78      0.77      0.78     27394
           2       0.47      0.48      0.48     11541

    accuracy                           0.69     38935
   macro avg       0.63      0.63      0.63     38935
weighted avg       0.69      0.69      0.69     38935
```


Saving completed

Using Random Forest Classifier

```
[0]: from sklearn.ensemble import RandomForestClassifier

random_forest = RandomForestClassifier(n_estimators=100, max_depth=2, random_state=0)

random_forest.fit(X_train,y_train)
score_2 = random_forest.score(X_test, y_test)
print(score_2)


plot_confusion_matrix(random_forest,X_test,y_test)


y_pred_2 = random_forest.predict(X_test)

print(classification_report(y_test, y_pred_2))
```
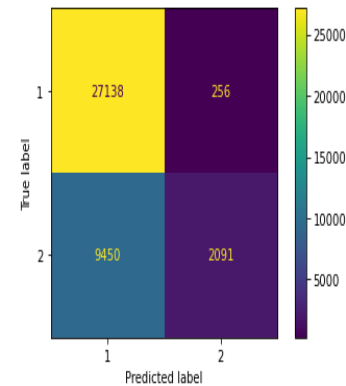
```
0.7507127263387697
              precision    recall  f1-score   support

           1       0.74      0.99      0.85     27394
           2       0.89      0.18      0.30     11541

    accuracy                           0.75     38935
   macro avg       0.82      0.59      0.57     38935
weighted avg       0.79      0.75      0.69     38935
```

Results and Evaluations

Conclusion

Based on the dataset provided for this capstone from weather, road, and light conditions pointing to certain classes, we can conclude that particular conditions have a somewhat impact on whether or not travel could result in property damage (class 1) or injury (class 2)