# Applied Data Science Capstone Project Report –

# Car accident severity

## 1. Introduction | Business Problem

In order to reduce the frequency of car collisions in a community, an algorithm is recommended to predict the severity of an accident given the key parameters such as current weather, road and visibility conditions. When conditions are bad, this model will alert drivers to remind them to be more careful.

## 2. Data Understanding

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

Our predictor is 'SEVERITYCODE'. It is used measure the severity of an accident from 0 to 4 within the dataset. Our attributes are used to weigh the severity of an accident are 'WEATHER', 'ROADCOND' and 'LIGHTCOND'.

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

## 3. Data Preprocessing

The 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.

```
print(final_df.round(0).astype(np.int64))
```

| | SEVERITYCODE | WEATHER | ROADCOND | LIGHTCOND |
|---|---|---|---|---|
| 25055 | 1 | 2 | 2 | 2 |
| 65280 | 1 | 1 | 1 | 1 |
| 86292 | 1 | 4 | 3 | 3 |
| 155111 | 1 | 1 | 1 | 1 |
| 64598 | 1 | 1 | 1 | 1 |

```
final_df.dtypes
```

```
SEVERITYCODE    int64
WEATHER         int64
ROADCOND        int64
LIGHTCOND       int64
```

# 4. Balancing the Dataset

Our target variable SEVERITYCODE is only 42% balanced. SEVERITYCODE in class 1 is nearly three times the size of class 2.

We can fix this by down sampling the majority class.

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

# 5. Methodology

Our data is now ready to be fed into machine learning models.
We will use the following models:

**K-Nearest Neighbor (KNN)**

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

**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.

**Logistic 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 good to use with logistic regression.

# 6. Results and Evaluation

Let's check the accuracy of our models.

```
# KNN Evaluation
```

```
from sklearn.metrics import jaccard_similarity_score
from sklearn.metrics import f1_score
from sklearn.metrics import log_loss
```

```
jaccard_similarity_score(y_test, knn_y_pred)
```

```
0.5186490455212922
```

```
f1_score(y_test, knn_y_pred, average='macro')
```

```
0.5180780960740707
```

```
# Decision Tree Evaluation
```

```
jaccard_similarity_score(y_test, dt_y_pred)
```

```
0.5572393538913363
```

```
f1_score(y_test, dt_y_pred, average='macro')
```

```
0.48610939554341154
```

```
LR_y_prob = LR.predict_proba(x_test)
log_loss(y_test, LR_y_prob)
```

```
0.6818543944060274
```

```
# Linear Regression Evaluation
```

```
jaccard_similarity_score(y_test, LR_y_pred)
```

```
0.5385315712187959
```

```
f1_score(y_test, LR_y_pred, average='macro')
```

```
0.5321816635757292
```

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

| ML Model | Jaccard Score | F1 Score | Log Loss |
|---|---|---|---|
| KNN | 0.52 | 0.52 | |
| Decision Tree | 0.56 | 0.49 | |
| Linear Regression | 0.54 | 0.53 | 0.68 |

# 7. Discussion

In the beginning of this notebook, 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 int64; a numerical data type.

In addition to the data type issue, we were presented with another challenge - imbalanced data. As mentioned earlier, class 1 was nearly three times larger than class 2. The solution to this was down sampling the majority class with sklearn's resample tool. We down sampled to match the minority class exactly with 58,188 values each.

Once we analyzed and cleaned the data, it was then fed through three ML models:

- K-Nearest Neighbor
- Decision Tree
- Logistic Regression

Although the first two models worked well for this project, logistic regression made most sense because of its binary nature.

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 hyparameter C values helped to improve our accuracy to be the best possible.

# 8. Conclusion

Based on historical data from weather conditions pointing to certain classes, we can conclude that particular weather conditions have a somewhat impact on whether or not travel could result in property damage (class 1) or injury (class 2).

Thank you for reading!