Accident Severity Machine Learning Models Based On Seattle Traffic Data

M. V. Tegethoff

September 2020

Outline

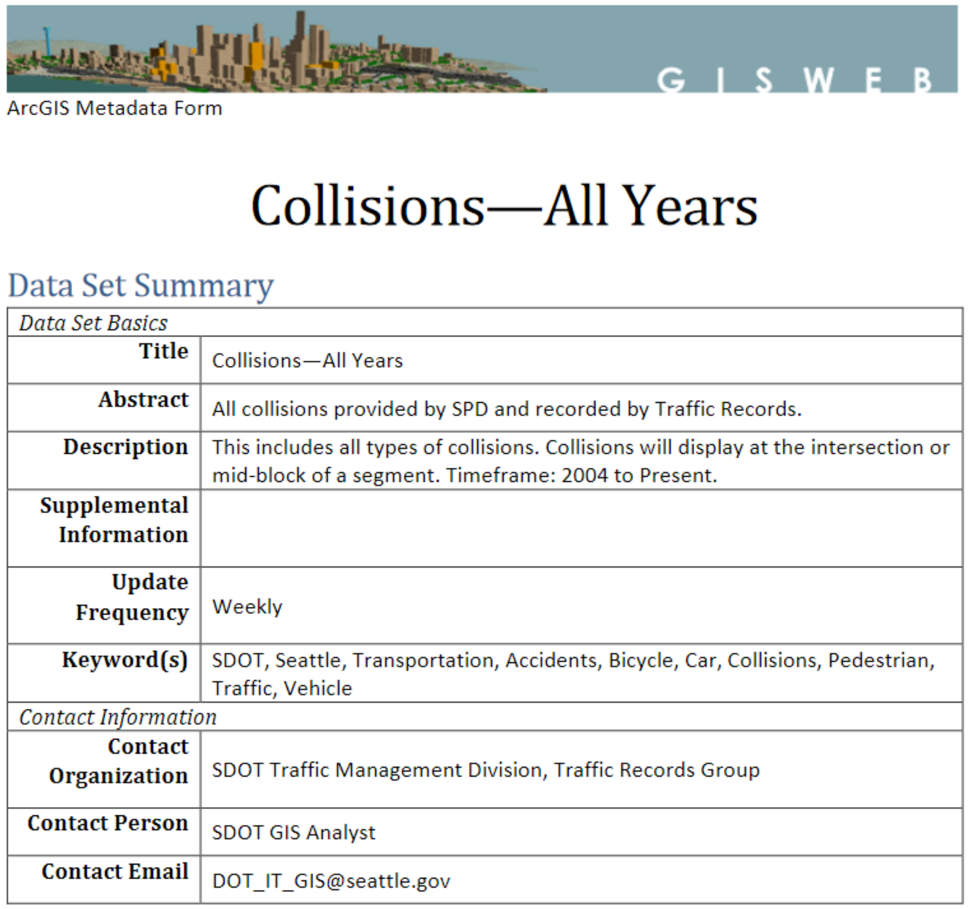
Introduction

- Reducing the number of accidents that result in injury is a critical objective for communities
- Data Science provides a proactive approach to reducing injuries
- Many sources of public data available documenting the severity of traffic accidents
- Machine learning models that can help predict the attributes that contribute to accident severity

This project examines a data set for the city of Seattle and develops several machine learning models that predict accident severity

The Data Set

- The data set used contains the city of Seattle collisions from 2006 to 2015
- Data is provided by SDOT Traffic Management Division, Traffic Records Group.
- The data set is available from Seattle Traffic Data Metadata and Seattle Traffic Dataset.
- Additional information is available in the 2015 Traffic Report by



ArcGIS Metadata Form

Collisions—All Years

Data Set Summary

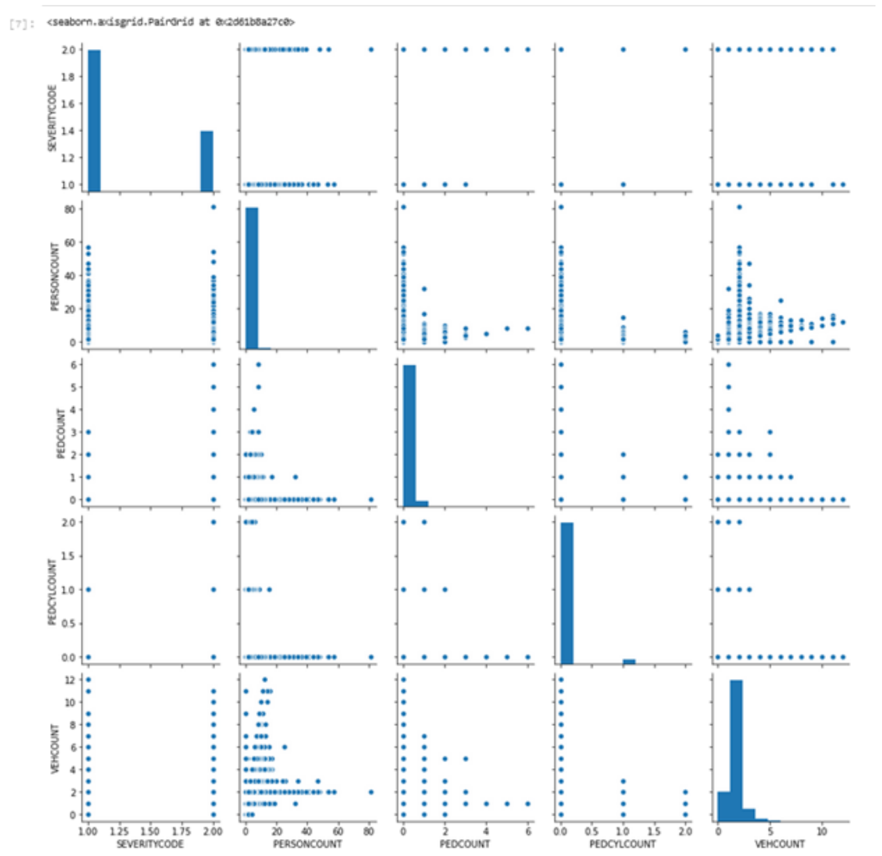
Data Set Basics	
Title	Collisions—All Years
Abstract	All collisions provided by SPD and recorded by Traffic Records.
Description	This includes all types of collisions. Collisions will display at the intersection or mid-block of a segment. Timeframe: 2004 to Present.
Supplemental Information	
Update Frequency	Weekly
Keyword(s)	SDOT, Seattle, Transportation, Accidents, Bicycle, Car, Collisions, Pedestrian, Traffic, Vehicle
Contact Information	
Contact Organization	SDOT Traffic Management Division, Traffic Records Group
Contact Person	SDOT GIS Analyst
Contact Email	DOT_IT_GIS@seattle.gov

Outline

Data Analysis and Attribute Selection

- The first step is to analyze the data set
  - Pair plot analysis was used to examine the data.
  - Removing non- significant attributes
  - Select the best attributes
- Examining individuals attribute.
- comparing attributes with collision severity.
- Attributes with impact include
  - Clear versus raining weather
  - Wet versus dry road condition
  - Day versus nigh light condition
- In Month the impact is uniform ally distributed

Pair Plot Analysis

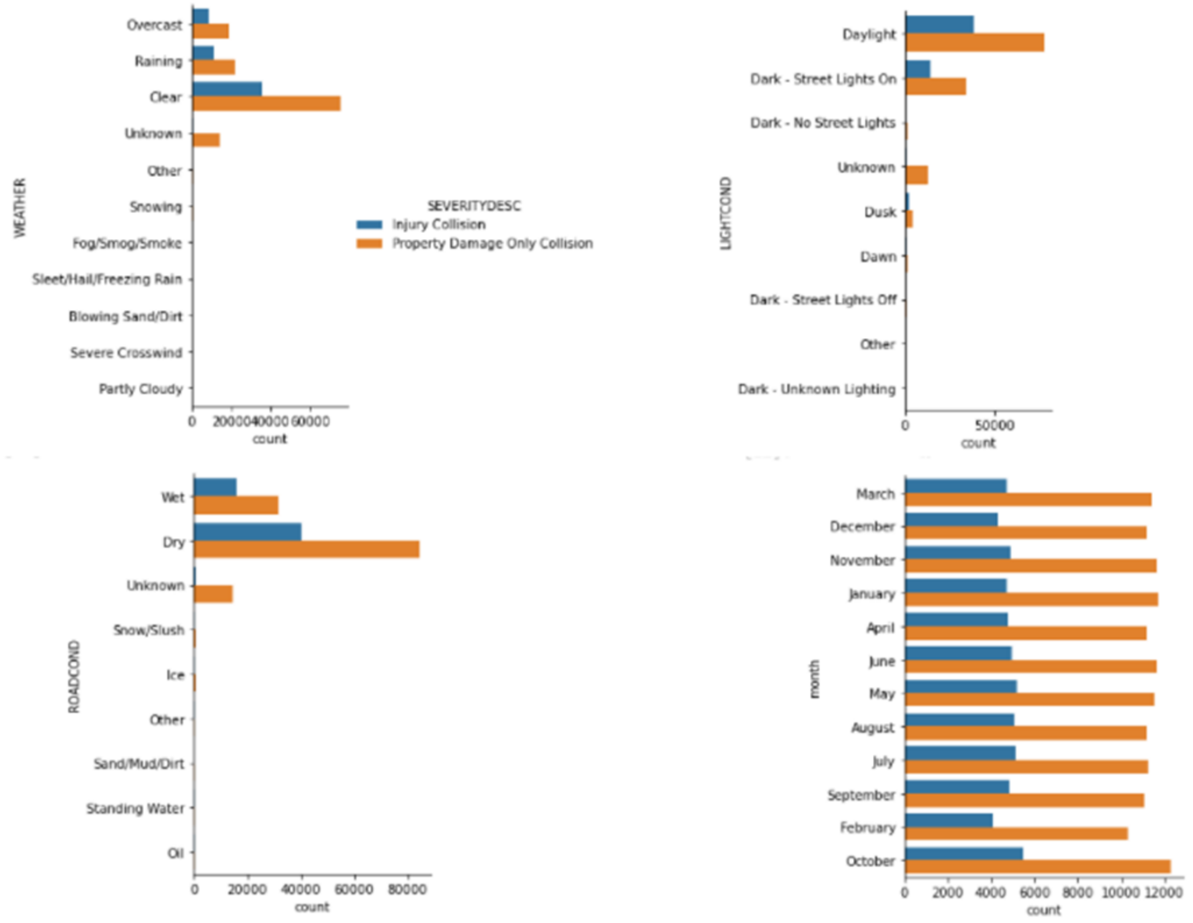


```
[7]: <seaborn.axisgrid.PairGrid at 0x2d61b8a27c0>
```

Data Analysis results

- Key attributes selected include
  - Weather
  - road condition
  - light condition
- Further combining of attributes
  - Increase the model performance efficiency by
  - Combining multiple selections into categories.
  - For example, combinations of “Dark” will be combined in a single “Dark” attribute.
- Note that the ratio of injury and non-injury accidents seems to remain somewhat constant across different road, light and weather conditions
- This will be further examines with a correlation matrix.

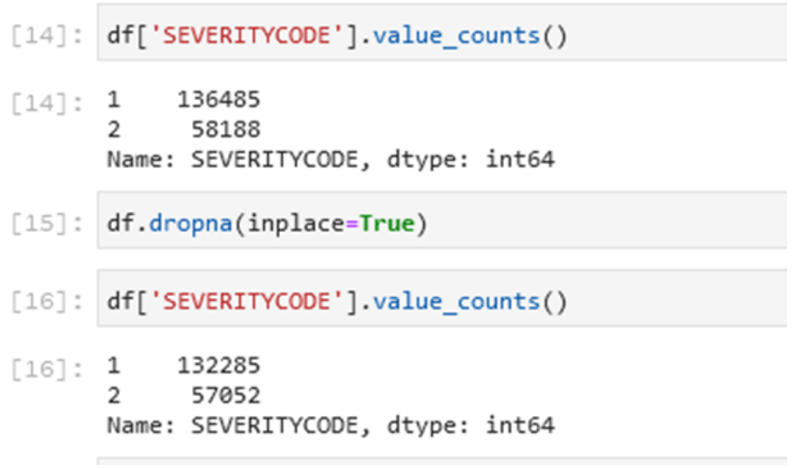
Histograms Attributes vs Collision Severity



Data Preparation

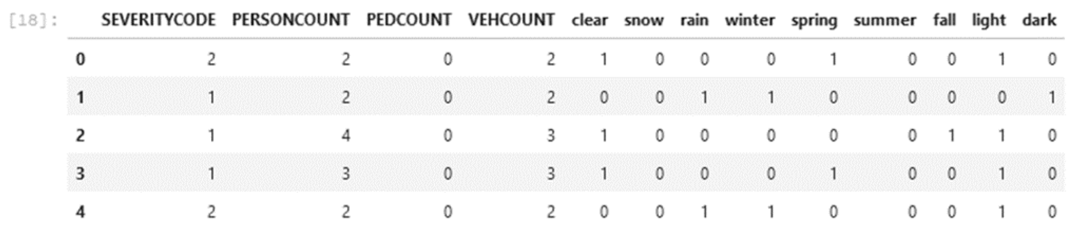
- Remove missing values
  - removed 4,200 non injury entries
  - 1,136 injury entries
- Convert Object to numerical
  - Use pd.get_dummies(df) to convert the object entries to numerical values
  - Further combined the weather into clear, rain and snow and light condition into light and dark

NULL value removal



```
[14]: df['SEVERITYCODE'].value_counts()

[14]: 1    136485
      2     58188
      Name: SEVERITYCODE, dtype: int64

[15]: df.dropna(inplace=True)

[16]: df['SEVERITYCODE'].value_counts()

[16]: 1    132285
      2     57052
      Name: SEVERITYCODE, dtype: int64
```

Data Frame after Attribute Selection



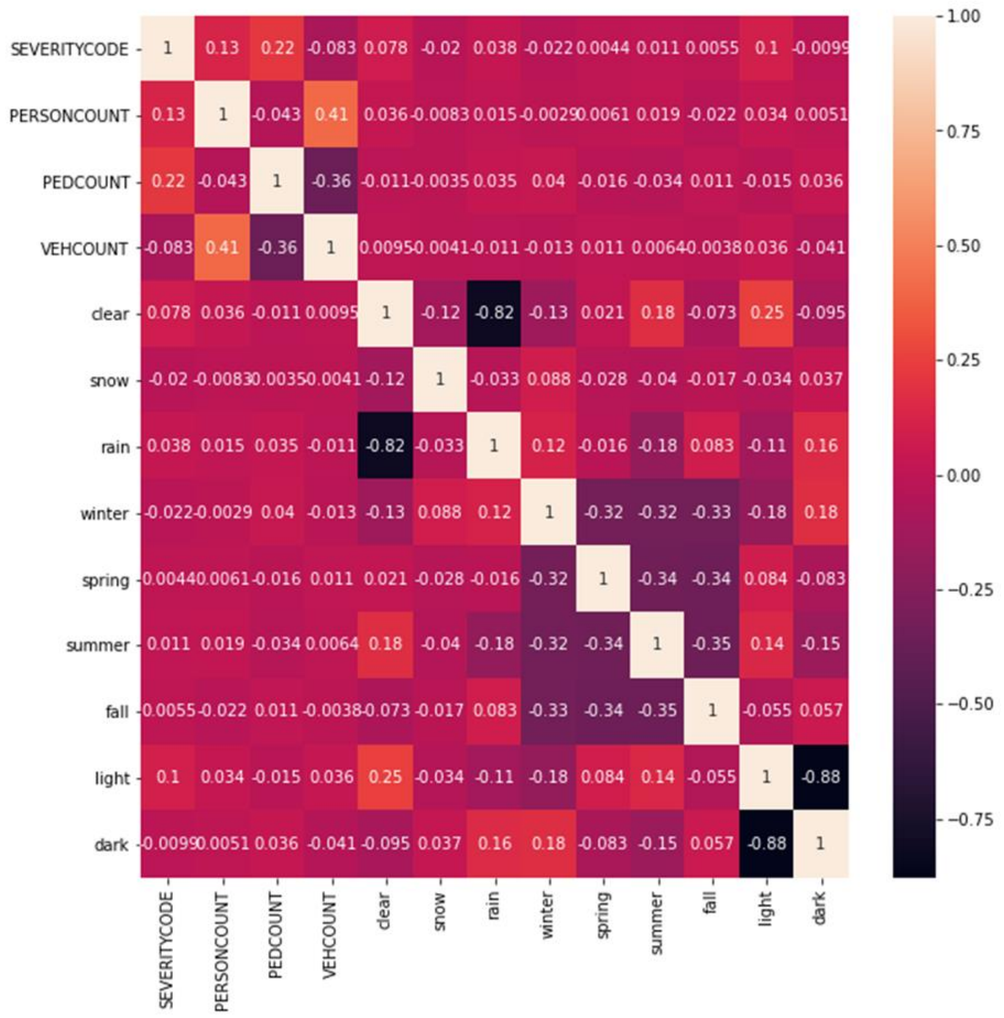
[18]:

	SEVERITYCODE	PERSONCOUNT	PEDCOUNT	VEHCOUNT	clear	snow	rain	winter	spring	summer	fall	light	dark
0	2	2	0	2	1	0	0	0	1	0	0	1	0
1	1	2	0	2	0	0	1	1	0	0	0	0	1
2	1	4	0	3	1	0	0	0	0	0	1	1	0
3	1	3	0	3	1	0	0	0	1	0	0	1	0
4	2	2	0	2	0	0	1	1	0	0	0	1	0

Correlation Heatmap

- As a result of the analysis, weather, road condition and light condition were selected as the key attributes for the modeling.
- Used correlation heatmap of the selected attributes.
- Correlation is useful to see if pairs of variables are related helping understand the data.
- Unfortunately the correlation matrix does not show strong correlation between any of the variables and the severity of the accident.

Attribute Correlation Heatmap

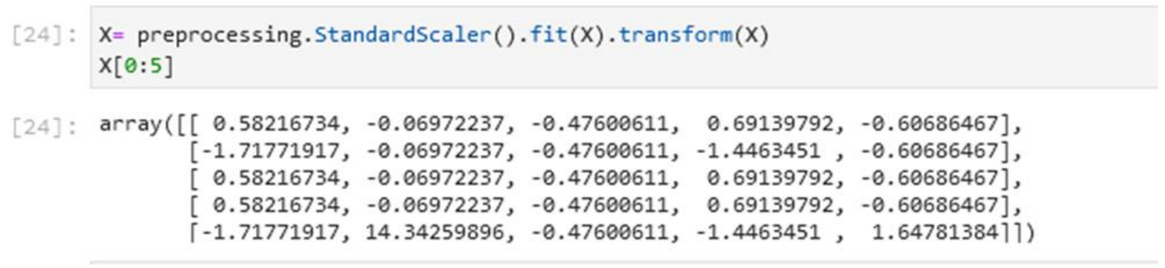


	SEVERITYCODE	PERSONCOUNT	PEDCOUNT	VEHCOUNT	clear	snow	rain	winter	spring	summer	fall	light	dark
SEVERITYCODE	1	0.13	0.22	-0.083	0.078	-0.02	0.038	-0.022	0.0044	0.011	0.0055	0.1	-0.0099
PERSONCOUNT	0.13	1	-0.043	0.41	0.036	-0.0083	0.015	-0.0029	0.0061	0.019	-0.022	0.034	0.0051
PEDCOUNT	0.22	-0.043	1	-0.36	-0.011	-0.0035	0.035	0.04	-0.016	-0.034	0.011	-0.015	0.036
VEHCOUNT	-0.083	0.41	-0.36	1	0.0095	-0.0041	-0.011	-0.013	0.011	0.0064	-0.0038	0.036	-0.041
clear	0.078	0.036	-0.011	0.0095	1	-0.12	-0.82	-0.13	0.021	0.18	-0.073	0.25	-0.095
snow	-0.02	-0.0083	-0.0035	-0.0041	-0.12	1	-0.033	0.088	-0.028	-0.04	-0.017	-0.034	0.037
rain	0.038	0.015	0.035	-0.011	-0.82	-0.033	1	0.12	-0.016	-0.18	0.083	-0.11	0.16
winter	-0.022	-0.0029	0.04	-0.013	-0.13	0.088	0.12	1	-0.32	-0.32	-0.33	-0.18	0.18
spring	0.0044	0.0061	-0.016	0.011	0.021	-0.028	-0.016	-0.32	1	-0.34	-0.34	0.084	-0.083
summer	0.011	0.019	-0.034	0.0064	0.18	-0.04	-0.18	-0.32	-0.34	1	-0.35	0.14	-0.15
fall	0.0055	-0.022	0.011	-0.0038	-0.073	-0.017	0.083	-0.33	-0.34	-0.35	1	-0.055	0.057
light	0.1	0.034	-0.015	0.036	0.25	-0.034	-0.11	-0.18	0.084	0.14	-0.055	1	-0.88
dark	-0.0099	0.0051	0.036	-0.041	-0.095	0.037	0.16	0.18	-0.083	-0.15	0.057	-0.88	1

Data Balancing and Transformation

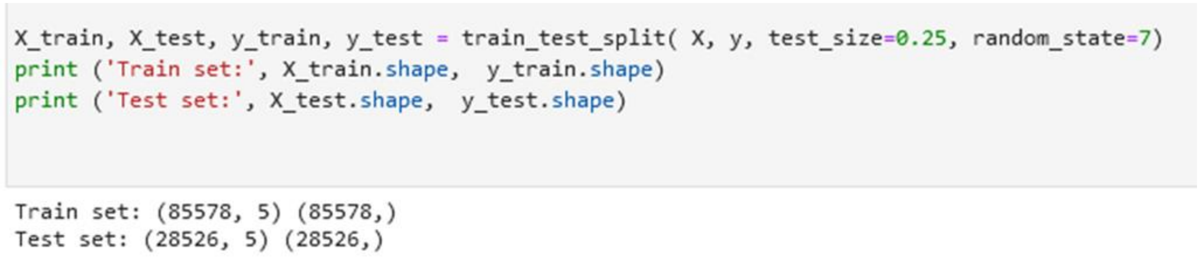
- The original dataset is unbalanced
  - 136,485 entries in the Property Damage class
  - 58,188 entries in the Injury collision class
- Two approaches were tried for balancing
  - Downsample the majority class and
  - Upsample minority class
  - Used Downsample the majority class with 57,052 for each class.
- Data tranformation
  - Transformed to make it look like standard normally distributed in a Gaussian manner with zero mean and unit variance
- Training and Testing separation
  - data was separated into training data and test data.
  - Several splits were considered
  - Final value was chosen to be 25% for test data.

Transformed Data



```
[24]: X= preprocessing.StandardScaler().fit(X).transform(X)
      X[0:5]

[24]: array([[ 0.58216734, -0.06972237, -0.47600611,  0.69139792, -0.60686467],
             [-1.71771917, -0.06972237, -0.47600611, -1.4463451 , -0.60686467],
             [ 0.58216734, -0.06972237, -0.47600611,  0.69139792, -0.60686467],
             [ 0.58216734, -0.06972237, -0.47600611,  0.69139792, -0.60686467],
             [-1.71771917, 14.34259896, -0.47600611, -1.4463451 ,  1.64781384]])
```

Train and Test Data



```
X_train, X_test, y_train, y_test = train_test_split( X, y, test_size=0.25, random_state=7)
print ('Train set:', X_train.shape,  y_train.shape)
print ('Test set:', X_test.shape,  y_test.shape)

Train set: (85578, 5) (85578,)
Test set: (28526, 5) (28526,)
```

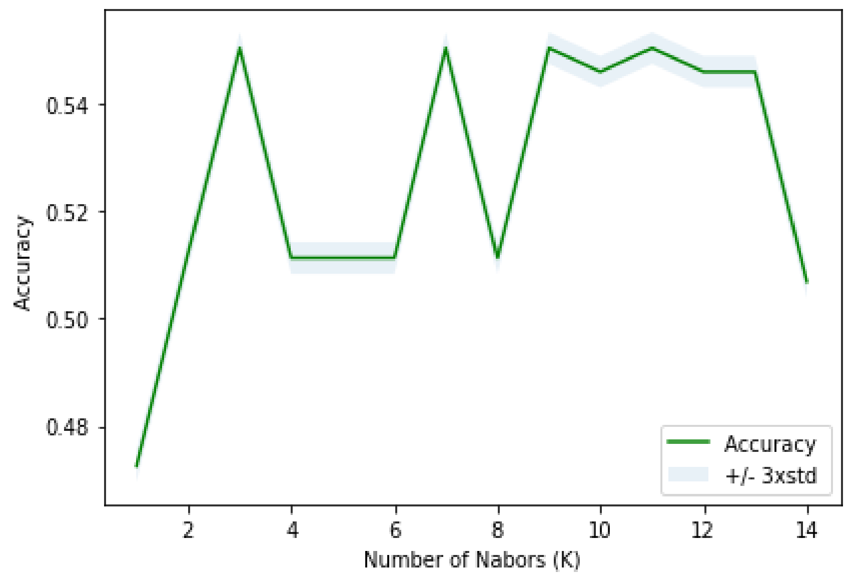
Outline

- Introduction
- Methodology
    - Data Analysis and Attribute Selection
    - Data Preparation
    - Data Balancing and Transformation
    - Data Training and Testing
- Model Implementation
    - KNN Model
    - Decision Tree Model
    - SVM Model
    - Logistic Regression Model
    - Random Forest Model
- Results and Discussion
- Conclusion

KNN Model

- The K-nearest neighbors (KNN)
  - Supervised machine learning algorithm
  - Can be used for classification tasks.
  - Does not make assumptions about the underlying data
  - Tried model for multiple values of K

KNN Results

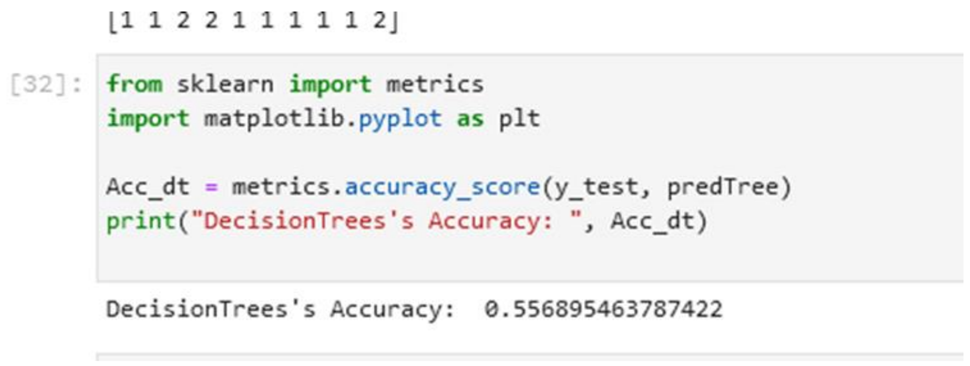


```
The best accuracy of KNN is  0.5503750964032812 , k= 3
KNN's Accuracy:  0.5503750964032812
```

Decision Tree

- Supervised learning method
- Used for classification and regression.
- The goal is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features.

Decision Tree Results

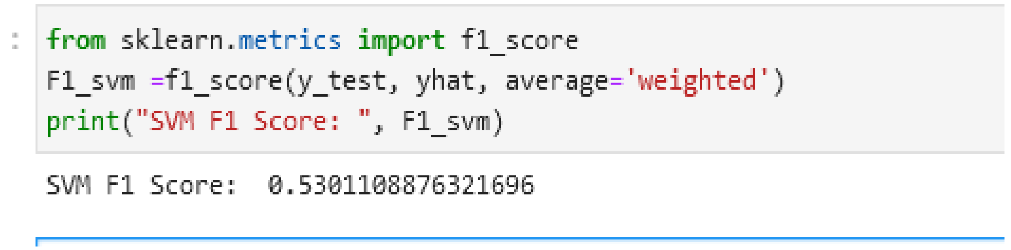


```
[1 1 2 2 1 1 1 1 1 2]

[32]: from sklearn import metrics
      import matplotlib.pyplot as plt

      Acc_dt = metrics.accuracy_score(y_test, predTree)
      print("DecisionTrees's Accuracy: ", Acc_dt)

      DecisionTrees's Accuracy:  0.556895463787422
```

SVM Model

- Support vector machines (SVMs)
- set of supervised learning methods used for classification, regression and outliers detection
- SVM is effective in high dimensional spaces.
- Different Kernel functions can be specified for the decision function.
- Common kernels are provided, but it is also possible to specify custom kernels.

SVM Results



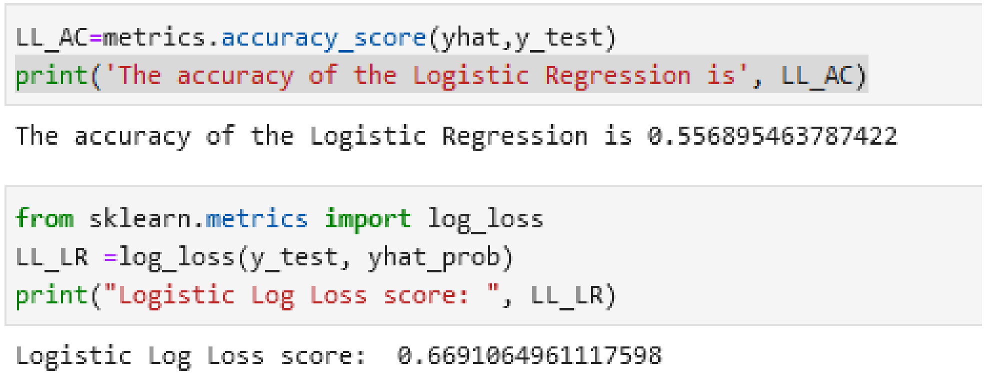
```
from sklearn.metrics import f1_score
F1_svm =f1_score(y_test, yhat, average='weighted')
print("SVM F1 Score: ", F1_svm)
```

```
SVM F1 Score:  0.5301108876321696
```

Logistic Regression Model

- Logistic Regression is one of the most simple and commonly used Machine Learning algorithms for two-class classification.
- It is easy to implement and can be used as the baseline for any binary classification problem.
- Its basic fundamental concepts are also constructive in deep learning.
- Logistic regression describes and estimates the relationship between one dependent binary variable and independent variables.

Logistic Regression Results



```
LL_AC=metrics.accuracy_score(yhat,y_test)
print('The accuracy of the Logistic Regression is', LL_AC)
```

```
The accuracy of the Logistic Regression is 0.556895463787422
```

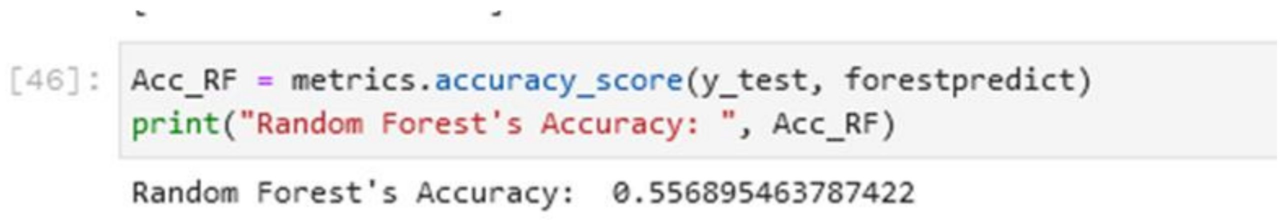
```
from sklearn.metrics import log_loss
LL_LR =log_loss(y_test, yhat_prob)
print("Logistic Log Loss score: ", LL_LR)
```

```
Logistic Log Loss score:  0.6691064961117598
```

Random Forest Model

- Random forests is a supervised learning algorithm
- Used both for classification and regression
- It creates decision trees on randomly selected data samples, gets prediction from each tree and selects the best solution by means of voting.

Random Forest Results



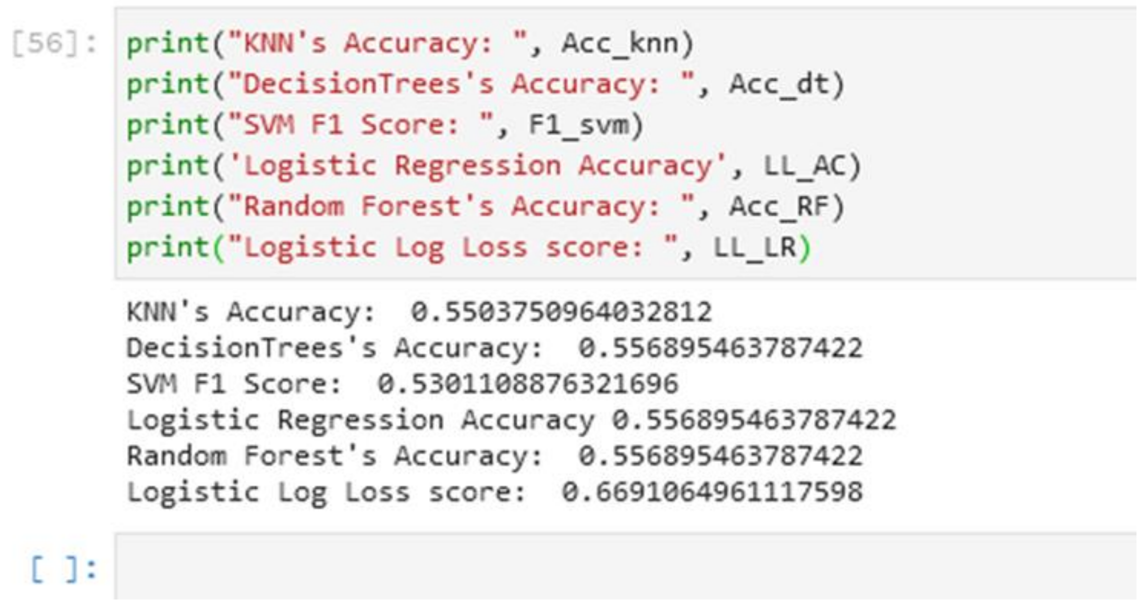
```
[46]: Acc_RF = metrics.accuracy_score(y_test, forestpredict)
      print("Random Forest's Accuracy: ", Acc_RF)

Random Forest's Accuracy:  0.556895463787422
```

Outline

Results and Discussion

- One would expect that weather, light condition and road condition would have an impact on severity of accidents.
-
- However, the accuracy of the models we generated is less that we would want.
- That was expected based on what we saw in the correlation analysis.
- In terms of comparing the different models, all models show similar accuracy around 0.55 except for Logistic regression.

Model Results



```
[56]: print("KNN's Accuracy: ", Acc_knn)
      print("DecisionTrees's Accuracy: ", Acc_dt)
      print("SVM F1 Score: ", F1_svm)
      print('Logistic Regression Accuracy', LL_AC)
      print("Random Forest's Accuracy: ", Acc_RF)
      print("Logistic Log Loss score: ", LL_LR)

      KNN's Accuracy:  0.5503750964032812
      DecisionTrees's Accuracy:  0.556895463787422
      SVM F1 Score:  0.5301108876321696
      Logistic Regression Accuracy 0.556895463787422
      Random Forest's Accuracy:  0.556895463787422
      Logistic Log Loss score:  0.6691064961117598

[ ]:
```

Conclusion

- In this project we used a data set for the city of Seattle to develop several machine learning model that predict accident severity. Accident severity is defined as the understanding if traffic accidents result in injury or property damage.
- Reducing the number of accidents that result in injury is a critical objective for communities, traffic engineers and the automotive industries.
- The methodology used included data analysis, attribute selection, data preparation and data balancing. KNN, Decision Tree, SVM, Logistic Regression, and Random Forest were developed to predict accident severity and their accuracy estimated using various metrics.
- The accuracy of the models we generated is less that we would want an the correlation between the available attributes and accident severity was not strong.
- In terms of comparing the different models, all models show similar accuracy around 0.55 except for Logistic regression that shows a better showing in terms of log loss accuracy.

References

- [1] https://scikit-learn.org/stable/modules/svm.html
- [2] https://www.datacamp.com/