Prediction of Car Accident Severity

By Ayush Rajpal

Applied Data Science Capstone

IBM Data Science Professional Certificate

Outline

- Introduction
- Data
- Methodology
- Results
- Discussion
- Conclusion

Introduction

▶ Almost 40,000 people are killed each year in the US due to traffic fatalities (https://en.wikipedia.org/wiki/Motor_vehicle_fatality_rate_in_U.S._by_year) .

▶ To better understand the reasons behind those accidents a data science study into the attributes leading to those accidents is warranted.

▶ Will be evaluating Seattle traffic accidents for past 15 years.

▶ This study will be of interest to:

▶ rivers and pedestrians in the greater Seattle metropolitan area

▶ edestrians

▶ city planners

▶ emergency responders

▶ road construction crews

▶ insurance companies.

Data

- Study data comes from Seattle Police Department:
  https://s3.us.cloud-object-storage.appdomain.cloud/cf-courses-data/CognitiveClass/DP0701EN/version-2/Data-Collisions.csv,
  - Records from 2004 to present updated weekly.
  - Data stored in a .csv file

- Study data includes the following information:
  - The severity of the accident
  - Type of collision
  - number of fatalities and/or injuries
  - Weather conditions
  - Road conditions
  - Any pedestrians or non-automobiles involved
  - And other factors.

- Metadata explaining these attributes is located at:

- https://s3.us.cloud-object-storage.appdomain.cloud/cf-courses-data/CognitiveClass/DP0701EN/version-2/Metadata.pdf.
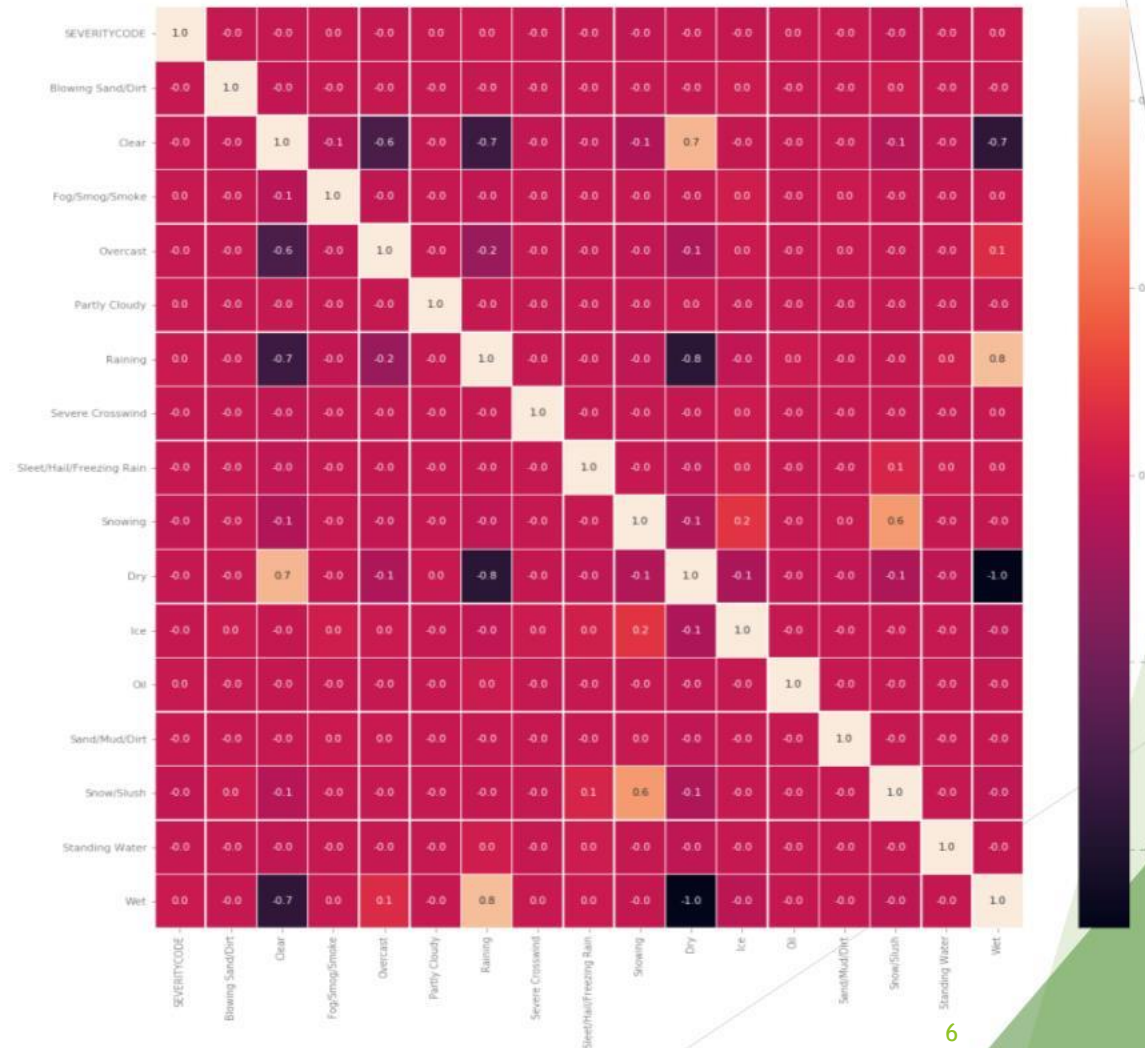
Methodology

Cleaning

▶ The raw dataset has 194673 rows and 38 columns.

▶ Duplicate columns were dropped.

▶ Unclear values were set to ⬜ NaN and later dropped.

▶ Categorical variables were converted to Numerical variables.

▶ Several variable sets (i.e. WEATHER, ROADCOND, LIGHTCOND) were split into dummy variables using their underlying characteristics.

▶ Final cleaned dataset has 194673 rows and 35 features.

Methodology
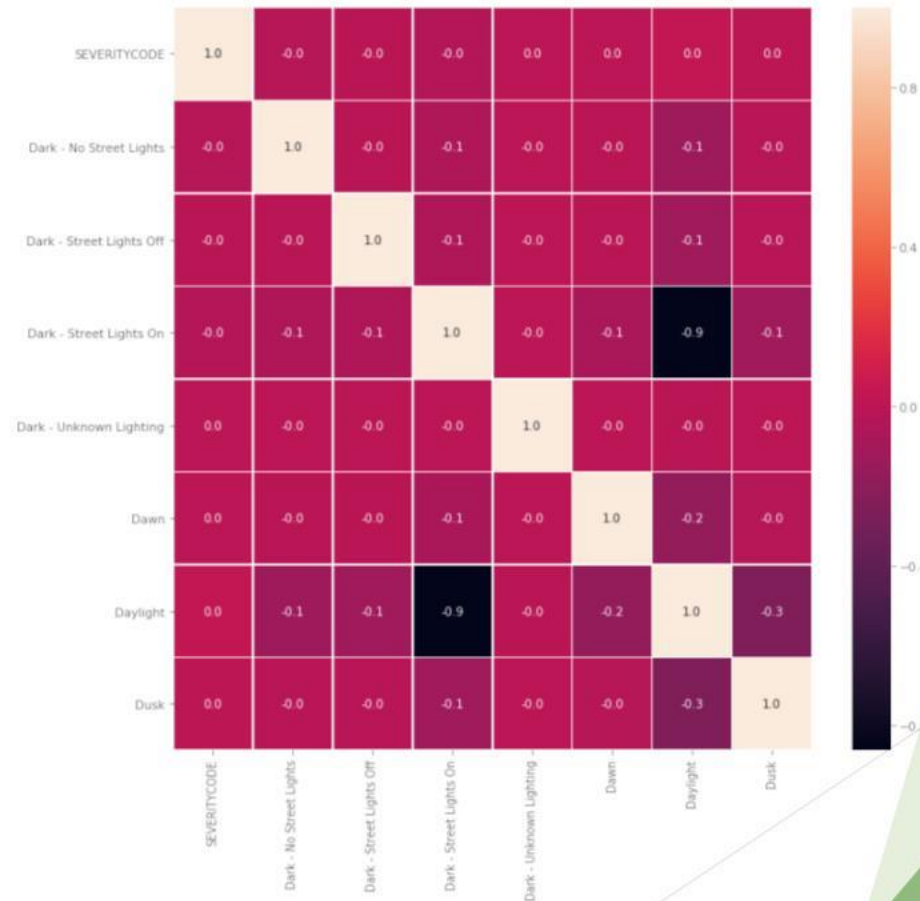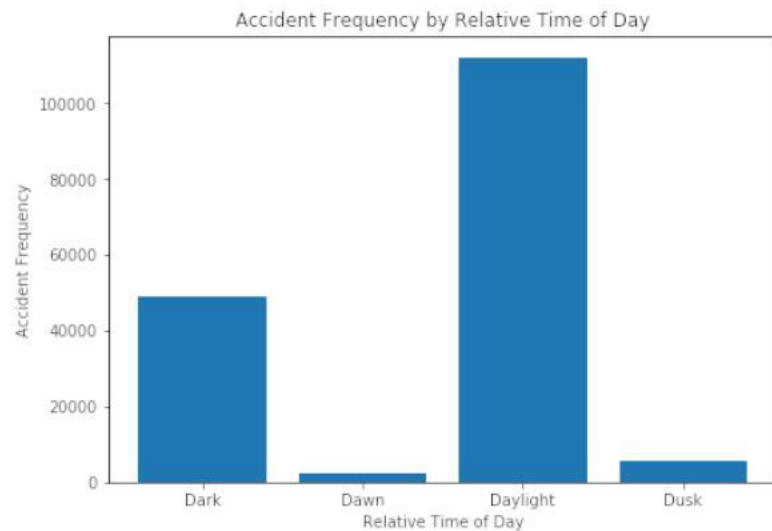## Analysis – Weather vs. Accident Severity

▶ No correlation between weather or road conditions and accident severity.

Methodology
Analysis – Lighting vs. Accident Severity

- No correlation between lighting conditions and accident severity.

- Most accidents occur during daytime.

Accident Frequency by Relative Time of Day

Methodology

Analysis –Impairment/Attention vs. Accident Severity

▶ No correlation between driver impairment or attention and accident severity.
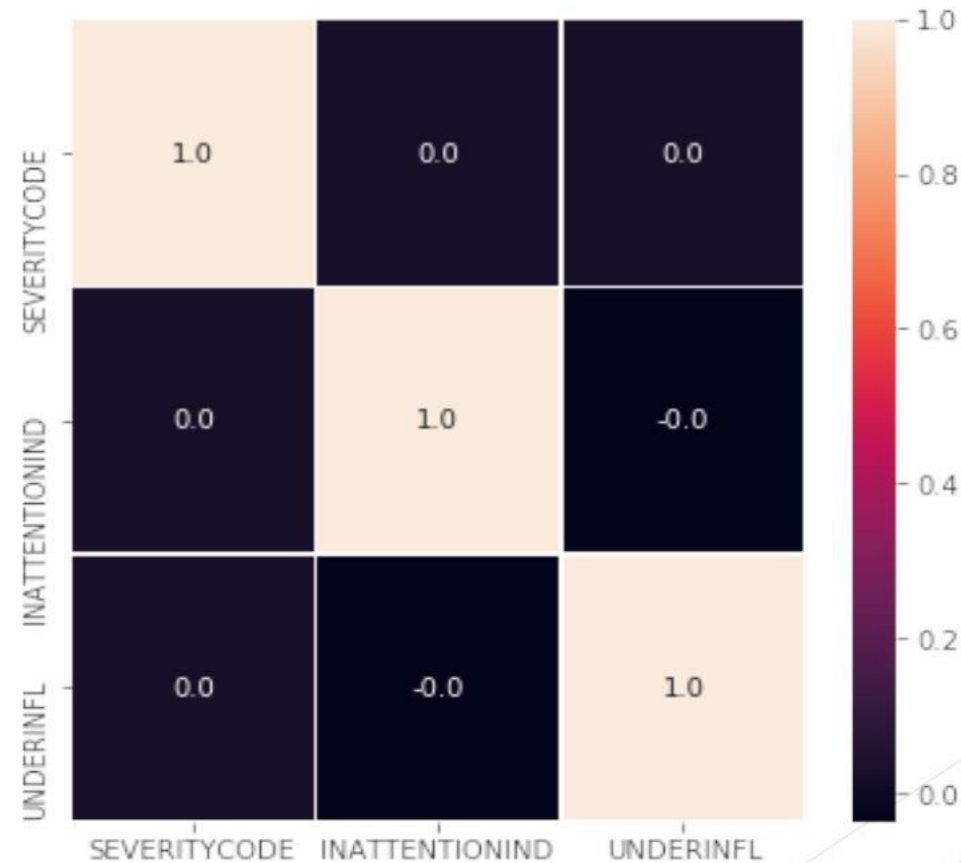
Methodology
Analysis – Location vs. Accident Severity

Out[23]:

	Accident Location	Severity = 1	Severity = 2	% Difference btwn Sev=1, Sev=2
0	Alley	516	77	0.850775
1	Intersection	34463	26808	0.222122
2	Block	78726	28657	0.635991

▶ Correlation between intersection accidents and accident severity.

Methodology
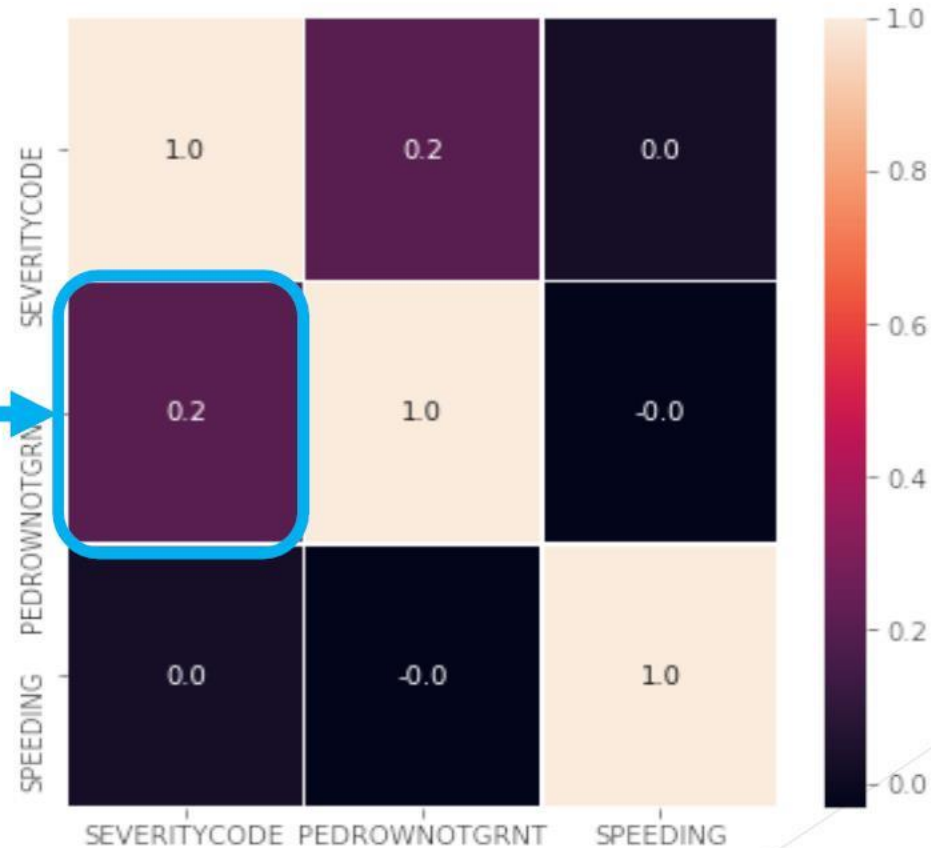## Analysis – Pedestrians vs. Accident Severity

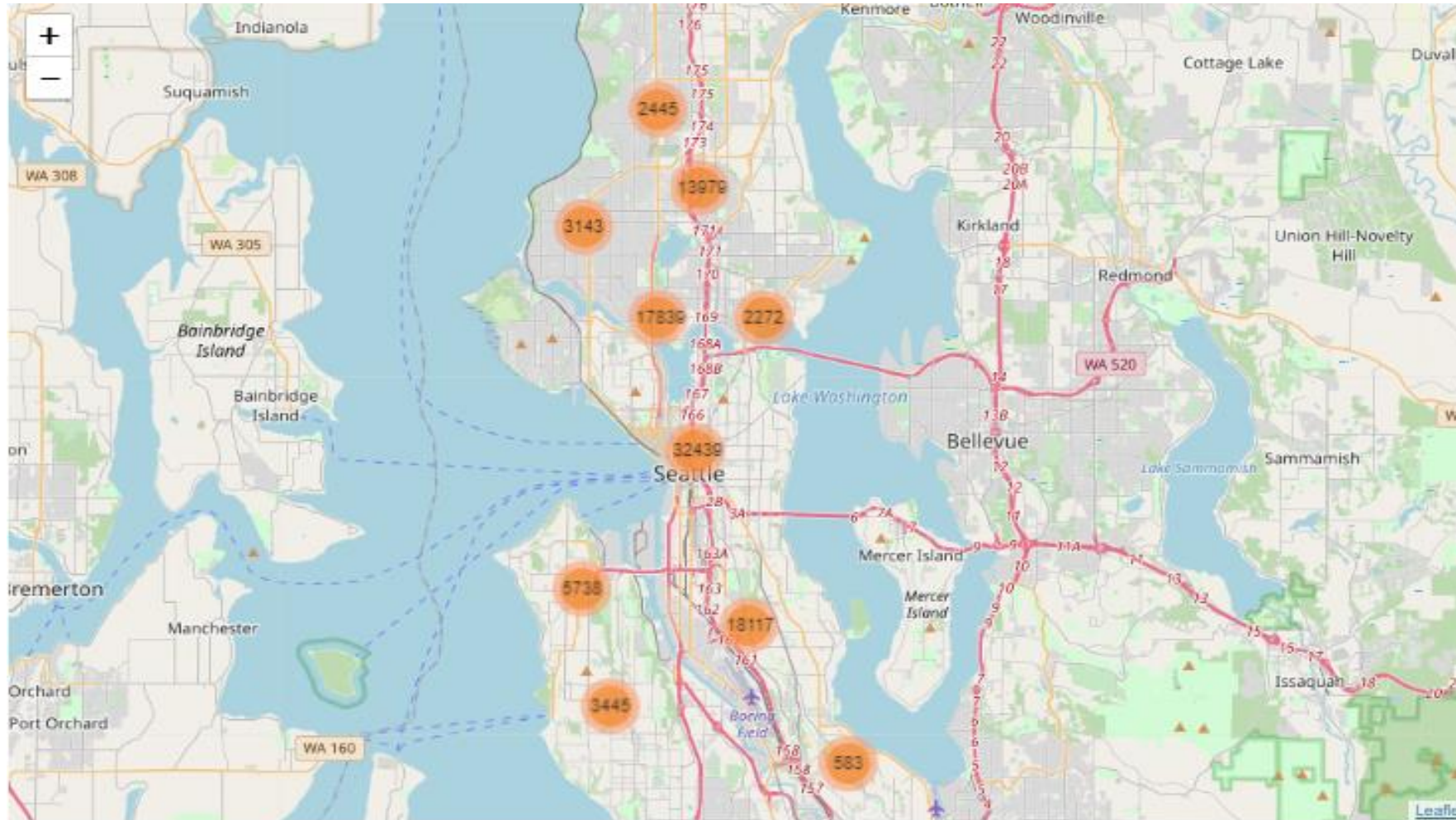▶ Correlation between the number of pedestrians and/or bicyclists and accident severity.

Methodology
Analysis – Speeding / Pedestrian ROW

▶ Correlation between pedestrian failed to be yielded right of way and accident severity.

Methodology
Map Showing Location of Accidents

Machine Learning Models
Decision Trees

▶ Decision Trees – depth – 8 gives best accuracy values.

```
In [31]:  # Create Jaccard and F1 Score Objects
          depthRange = range(1, 9)
          jaccardSimilarityScore = []
          f1Score = []

          # For Varying Depths, calculate jaccard similarity and f1 scores
          for n in depthRange:
              dTree = DecisionTreeClassifier(criterion = 'gini', max_depth = n)
              dTree.fit(X_train, y_train)
              dTree_yhat = dTree.predict(X_test)
              jaccardSimilarityScore.append(jaccard_similarity_score(y_test, dTree_yhat))
              f1Score.append(f1_score(y_test, dTree_yhat, average = 'weighted'))
```

```
In [32]:  # Present the resulting data in a DataFrame
          dTree_result = pd.DataFrame([jaccardSimilarityScore, f1Score],
                                      index = ['Jaccard Sim', 'F1'], columns = ['d = 1','d = 2','d = 3','d = 4','d = 5','d = 6','d = 7', 'd = 8'])
          dTree_result.columns.name = 'Depths'
          dTree_result
```

Out[32]:

Depths	d = 1	d = 2	d = 3	d = 4	d = 5	d = 6	d = 7	d = 8
Jaccard Sim	0.706470	0.729897	0.729897	0.729867	0.731640	0.732555	0.734032	0.735923
F1	0.616746	0.664202	0.664202	0.664151	0.675052	0.688503	0.684000	0.690943

Machine Learning Models

▶ Let's see the decision tree(click here)
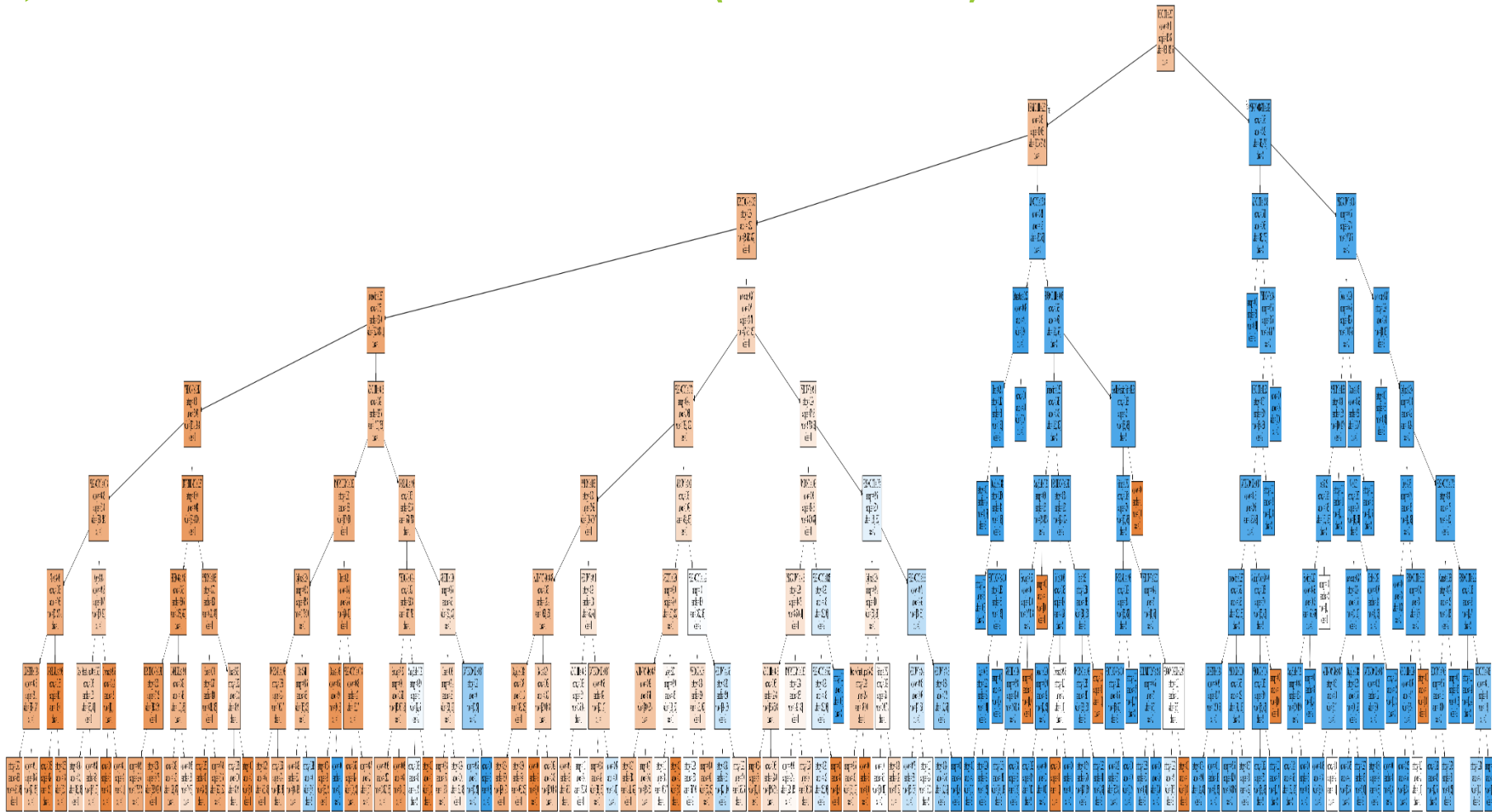
Machine Learning Models
Logistic Regression

► Logistic Regression – Lib-Linear Solver, Regularization Value = 0.01 give best accuracy values.

```
In [35]:  # Create Solver and Accuracy Score Obje
          solverList = ['lbfgs', 'saga', 'liblinear', 'newton-cg', 'sag']
          regularizationValueSet = [0.1, 0.01, 0.001]
          index = []
          lrAccuracy = []
          iterations = 0

          for p, q in enumerate(regularizationValueSet):
              for r, s in enumerate(solverList):
                  index.append(p + r *5)
                  iterations +=1
                  lrModel = LogisticRegression(C = q, solver = s)
                  lrModel.fit(X_train, y_train)
                  lr_yhat = lrModel.predict(X_test)
                  y_prob = lrModel.predict_proba(X_test)
                  print('Test {}: With C = {} for solver = {}, LR Accuracy is : {}'.format(iterations, q, s, log_loss(y_test, y_prob) ))
                  lrAccuracy.append(log_loss(y_test, y_prob))

              print('\n')
```

```
Test     With C = 0.1 for solver = lbfgs, LR Accuracy is : 0.5557660528012474
Test 2:  ith C = 0.1 for solver = saga, LR Accuracy is : 0.5557657197099635
Test 3: W   C = 0.1 for solver = liblinear, LR Accuracy is : 0.5557664324634823
Test 4: With    = 0.1 for solver = newton-cg, LR Accuracy is : 0.5557659873630711
Test 5: With C   0.1 for solver = sag, LR Accuracy is : 0.5557665902061812

Test 6: With C = 0.01 for solver = lbfgs, LR Accuracy is : 0.5557784963316049
Test 7: With C = 0.01 fo  solver = saga, LR Accuracy is : 0.5557786432657238
Test 8: With C = 0.01 for   lver = liblinear, LR Accuracy is : 0.5557831061701002
Test 9: With C = 0.01 for so er = newton-cg, LR Accuracy is : 0.5557784099363405
Test 10: With C = 0.01 for solv  = sag, LR Accuracy is : 0.5557783761980235

Test 11: With C = 0.001 for solver =    gs, LR Accuracy is : 0.5561091101536626
Test 12: With C = 0.001 for solver =       LR Accuracy is : 0.5561102050102353
Test 13: With C = 0.001 for solver = liblinear, LR Accuracy is : 0.5562142720546198
Test 14: With C = 0.001 for solver = newton-cg, LR Accuracy is : 0.5561095344008128
Test 15: With C = 0.001 for solver = sag, LR Accuracy is : 0.5561097202183379
```

Machine Learning Models Summary Statistics – Best model?

► According to the Jaccard Similarity Score, Decision Trees performed slightly better, though either technique could probably be used.

```
In [39]: # Decision Tree Model Scores
         f1_dTree = f1_score(y_test, dTree_yhat, average='weighted')
         jss_dTree = jaccard_similarity_score(y_test, dTree_yhat)
         ps_dTree = precision_score(y_test, dTree_yhat)
         rs_dTree = recall_score(y_test, dTree_yhat)

         #Logistic Regression Model Scores
         f1_lr = f1_score(y_test, lr_yhat, average='weighted')
         jss_lr = jaccard_similarity_score(y_test, lr_yhat)
         ps_lr = precision_score(y_test, lr_yhat)
         rs_lr = recall_score(y_test, lr_yhat)

         # Present Results
         ModelResults = {'Classification Model': ['Decision Trees', 'Logistic Regression'], 'f1 Score': [f1_dTree, f1_lr],
                         'Jaccard': [jss_dTree, jss_lr], 'Precision Score': [ps_dTree, ps_lr], 'Recall Score': [rs_dTree, rs_lr]};
         df_ModelResults = pd.DataFrame(ModelResults);
         df_ModelResults
```

Out[39]:

	Classification Model	f1 Score	Jaccard	Precision Score	Recall Score
0	Decision Trees	0.69094	0.735923	0.731966	0.959823
1	Logistic Regression	0.67879	0.732201	0.724842	0.971696

14

Results and Discussion

▶ Decision Tree: 73.6% match between training and test set.

▶ Logistic Regression: 73.2% match between training and test set.

▶ Weather conditions, road conditions, lighting conditions, driver impairment and speeding have little correlation with the severity of accident.

▶ Accident severity is increased in situations involving traffic intersections and the presence of pedestrians and/or bicyclists.

▶ Two main lessons from this project:

  ▶ Be very mindful of driving in intersections, as the likelihood of a severe accident is increased.

  ▶ When encountering pedestrians or bicyclists, extra caution should be taken.

Conclusions and Future Work

▶ To better understand the potential impacts (and possibly mitigate them), it is important to understand the attributes that contribute to severe accidents.

▶ There is no doubt that the information gathered here will be useful to drivers, pedestrians, city planners, emergency responders and insurance companies going forward.

▶ For future work, I would try the following:

▶ Evaluating the accuracy with other classification tools (i.e. K-Nearest Neighbor, Support Vector Machine) would be invaluable.

▶ Perhaps evaluating the make and model of the vehicles involved would also provide insight into traffic safety.

Thank you!