Seattle Car Collision Severity Prediction Model

September 12, 2020

Naveen Kumar Mangal

1. Introduction:

- Car accidents severity prediction is important to alert the people, improve driving conditions, and safe travels.

- Some bad conditions and wrong decisions can lead to road accidents of different severity.

- Predicting car collision severity will help drivers to:
  a) Provide assistance in decision making regarding travel conditions
  b) Help new driver (to a particular route) in selection of correct driving decisions
  c) Help insurance companies to predict the Collison severity to a particular area for assistance in claim settlement.

- Additionally, it help traffic police to manage the smooth traffic flow.

2. Data Acquisition and Cleaning

- We have selected weekly updated dataset by Seattle government, and the link of the dataset is following:

https://s3.us.cloud-object-storage.appdomain.cloud/cf-courses-data/CognitiveClass/DP0701EN/version-2/Data-Collisions.csv

- In total, it has shape of 194673 by 38.

- Down sampling was performed in SEVERITYCODE to balance the dataset.

3. Exploratory Dataset Analysis:

Table 1: Dataset Description

```
In [3]: df.describe()
```

Out[3]:

	SEVERITYCODE	X	Y	OBJECTID	INCKEY	COLDETKEY	INTKEY	SEVERITYCODE.1	PERSONCOUNT	PE
count	194673.000000	189339.000000	189339.000000	194673.000000	194673.000000	194673.000000	65070.000000	194673.000000	194673.000000	19467
mean	1.298901	-122.330518	47.619543	108479.364930	141091.456350	141298.811381	37558.450576	1.298901	2.444427	
std	0.457778	0.029976	0.056157	62649.722558	86634.402737	86986.542110	51745.990273	0.457778	1.345929	
min	1.000000	-122.419091	47.495573	1.000000	1001.000000	1001.000000	23807.000000	1.000000	0.000000	
25%	1.000000	-122.348673	47.575956	54267.000000	70383.000000	70383.000000	28667.000000	1.000000	2.000000	
50%	1.000000	-122.330224	47.615369	106912.000000	123363.000000	123363.000000	29973.000000	1.000000	2.000000	
75%	2.000000	-122.311937	47.663664	162272.000000	203319.000000	203459.000000	33973.000000	2.000000	3.000000	
max	2.000000	-122.238949	47.734142	219547.000000	331454.000000	332954.000000	757580.000000	2.000000	81.000000	

```
In [7]: import seaborn as sns
        import matplotlib.pyplot as plt

        plt.figure(figsize=(10,5))
        sns.heatmap(df.corr(),annot=True)

Out[7]: <matplotlib.axes._subplots.AxesSubplot at 0x1a21f70c9b0>
```

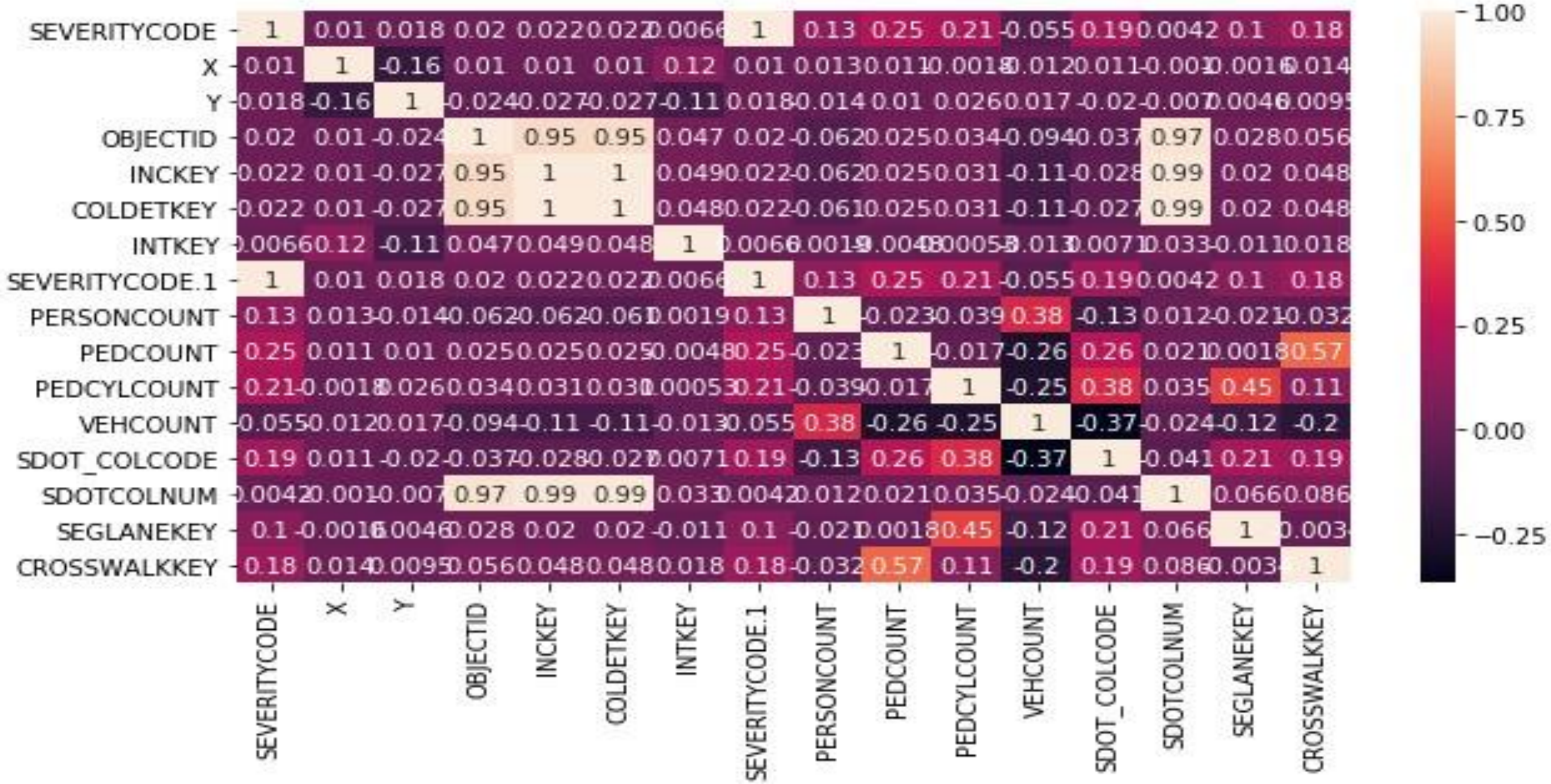

Fig 1: Dataset Heatmap

```
In [24]: df.plot(kind='box', figsize=(20, 8))

         plt.title('Box plot of Seattle Car Collision')
         plt.ylabel('Values')

         plt.show()
```

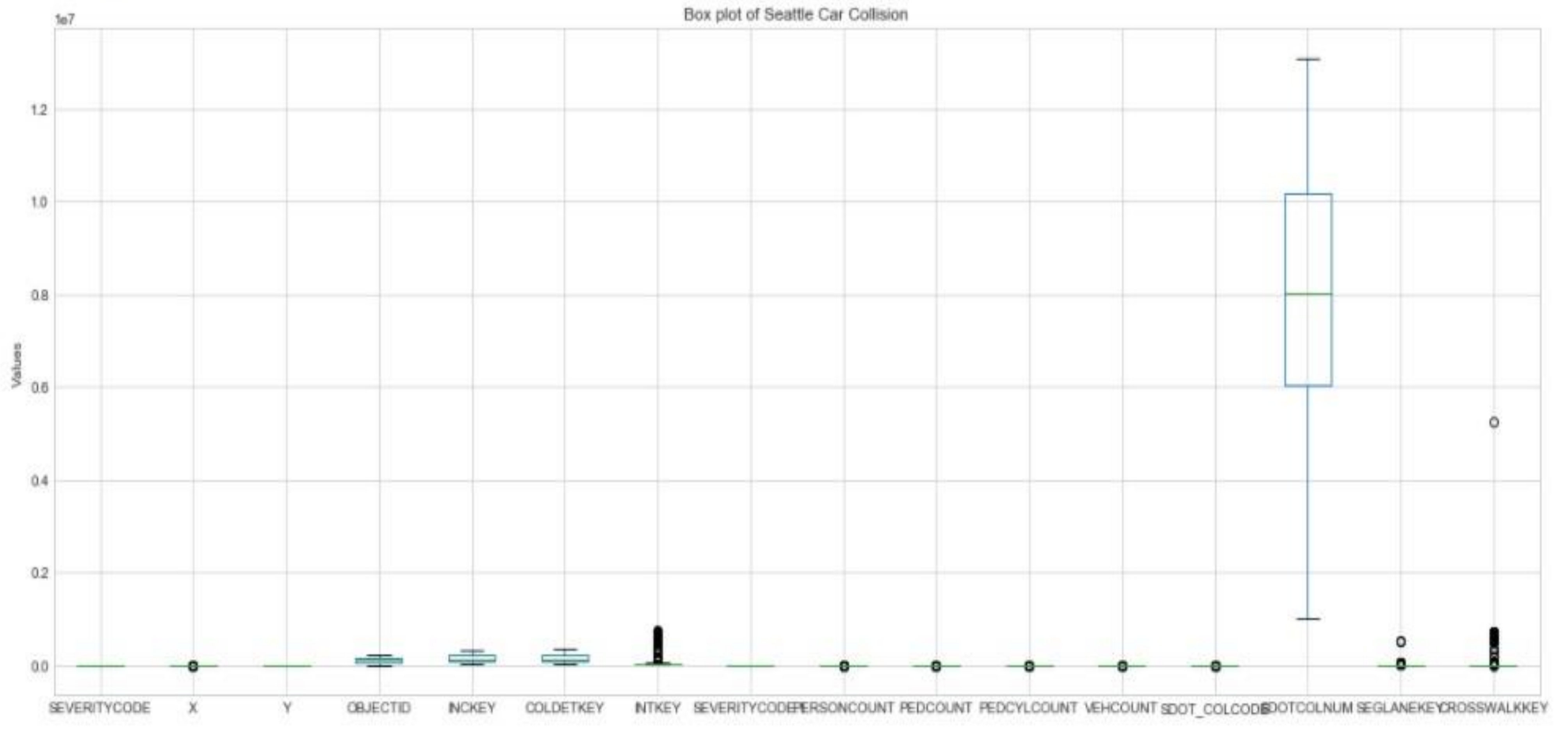

Fig 1: Dataset Boxplot

4. Machine Leaning models for prediction:

kNN ML Model

In [24]:
```
from sklearn.neighbors import KNeighborsClassifier
k=17
knn = KNeighborsClassifier(n_neighbors = k).fit(X_train,Y_train)
knn_Y_pred = knn.predict(X_test)
knn_Y_pred[0:5]
```

Out[24]:
```
array([2, 1, 2, 2, 1], dtype=int64)
```

Decision Tree ML Model

In [38]:
```
from sklearn.tree import DecisionTreeClassifier
dt = DecisionTreeClassifier(criterion = "entropy", max_depth = 7)
dt.fit(X_train,Y_train)
```

Out[38]:
```
DecisionTreeClassifier(class_weight=None, criterion='entropy', max_depth=7,
            max_features=None, max_leaf_nodes=None,
            min_impurity_decrease=0.0, min_impurity_split=None,
            min_samples_leaf=1, min_samples_split=2,
            min_weight_fraction_leaf=0.0, presort=False, random_state=None,
            splitter='best')
```

In [53]:
```
predTree = dt.predict(X_test)
print (predTree [0:5])
print (Y_test [0:5])
```

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

Logistic Regression ML Model

In [46]:
```
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import confusion_matrix
LR = LogisticRegression(C=6, solver='liblinear').fit(X_train,Y_train)
LR
```

Out[46]:
```
LogisticRegression(C=6, class_weight=None, dual=False, fit_intercept=True,
          intercept_scaling=1, max_iter=100, multi_class='ovr', n_jobs=1,
          penalty='l2', random_state=None, solver='liblinear', tol=0.0001,
          verbose=0, warm_start=False)
```

In [49]:
```
LRyhat = LR.predict(X_test)
LRyhat
```

Out[49]:
```
array([2, 2, 2, ..., 2, 2, 2], dtype=int64)
```

In [50]:
```
yhat_prob = LR.predict_proba(X_test)
yhat_prob
```

Out[50]:
```
array([[0.47215715, 0.52784285],
       [0.43603401, 0.56396599],
       [0.46429576, 0.53570424],
       ...,
       [0.47215715, 0.52784285],
       [0.46785336, 0.53214664],
       [0.47215715, 0.52784285]])
```

5. ML models evaluations:

Model	Jaccard Score	F1 Score	Accuracy
kNN	0.560	0.514	0.560
Decision Tree	0.563	0.532	0.563
Logistic Regression	0.53	0.517	0.532

Above table shows that, Decision tree model gives best predictions.

6. Discussion and Conclusion:

- Initially, we have object type values in selected attributes, so we have converted them into categorical type values.
- Down sampling of target variable to balance the dataset.
- Data was then fed through three ML models; K-Nearest Neighbor, Decision Tree and Logistic Regression.
- Evaluation metrics used for our models were **jaccard index, f-1 score, and accuracy.**
- 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).