The IBM Professional Data Science Capstone Project

Severity Code on car accidents Analysis

by Oriol Jorba Ferro

Introduction

- The world suffers due to car accidents, including the USA. National Highway Traffic Safety Administration of the USA suggests that the economical and societal harm from car accidents can cost up to $871 billion in a single year. The project aims to understand, study and predict the severity of accidents and the external factors that can play a role such as the weather, the road and light conditions.
- This model can be used for alerting drivers when the probability of having a severe accident is high, due to weather and other external conditions.

Data Understanding : Raw data / Target var.

- The raw data consist on all type of collisions on the city of Seattle from 2004 to present. It has 194673 observations (collisions).
- Target variable : **<u>Severity Code</u>** (binary variable)
  - Property damage only : y = 0*
  - Some type of injury : y = 1*

```
corr['SEVERITYCODE'][np.abs(corr['SEVERITYCODE'])>0.056]
```

```
8]:  SEVERITYCODE        1.000000
     X                   0.067429
     Y                   0.067592
     ADDRTYPE            0.172032
     INTKEY             -0.124089
     LOCATION            0.067601
     EXCEPTRSNDESC       0.078302
     SEVERITYCODE.1      1.000000
     SEVERITYDESC        1.000000
     COLLISIONTYPE      -0.211465
     PERSONCOUNT        -0.112706
     PEDCOUNT           -0.246338
     PEDCYLCOUNT        -0.214218
     VEHCOUNT           -0.181422
     SDOT_COLCODE       -0.072647
     SDOT_COLDESC       -0.072647
     WEATHER             0.056842
     ROADCOND            0.076572
     LIGHTCOND           0.084048
     PEDROWNOTGRNT      -0.206283
     ST_COLDESC         -0.157668
     SEGLANEKEY         -0.127947
     CROSSWALKKEY       -0.148796
     HITPARKEDCAR        0.101498
```

- Which predictor variables use?

Using a chi-distribution we have analysed the correlation between all variables and the target variable.

There is not an important relation between any variable, but there are some with $corr > 0.15$

Let's use them !

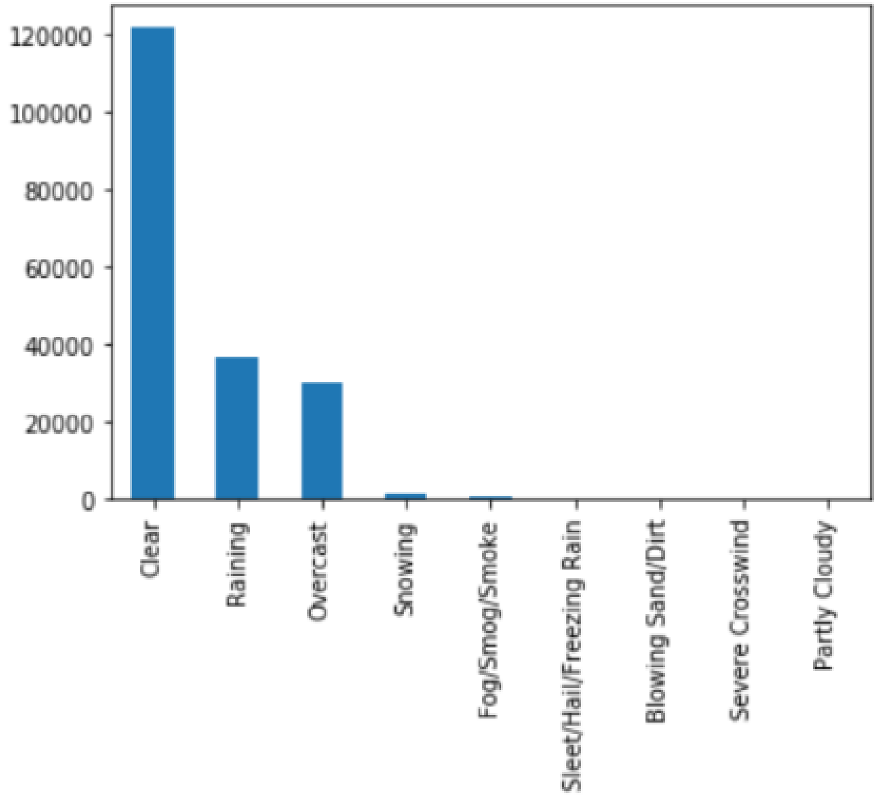
* In the raw data are prop. Dmg. 0, and inj. dmg. 1. We modified them for simplicity.

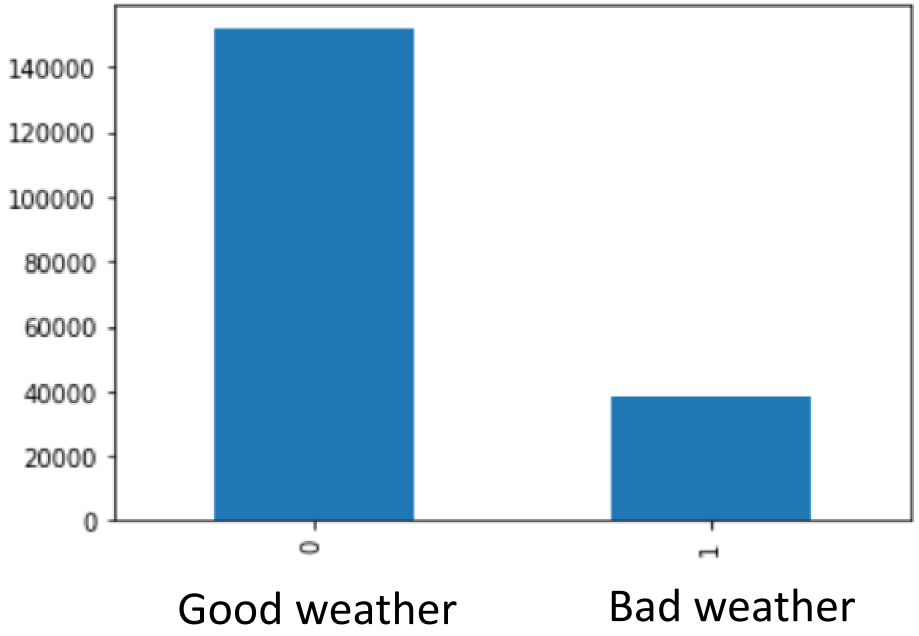
Data Understanding: Predictor Variables (I)

- **Root variables (factors that can lead to an accident):**
  - Weather
  - Road conditions
  - Light conditions,
  - Speeding,
  - Under influence of drugs/alcohol
  - Innatention to the road
  - Whether the pedestrian right of way was not granted.
- **Other variables:**
  - Address type
  - Collision type
  - Person count
  - Pedestrian count
  - Vehicle count
  - Number of Bicycles involved.

Data Understanding: Predictor Variables (II)

- We simplify most categorical variables into binary variables, e.g. Weather:

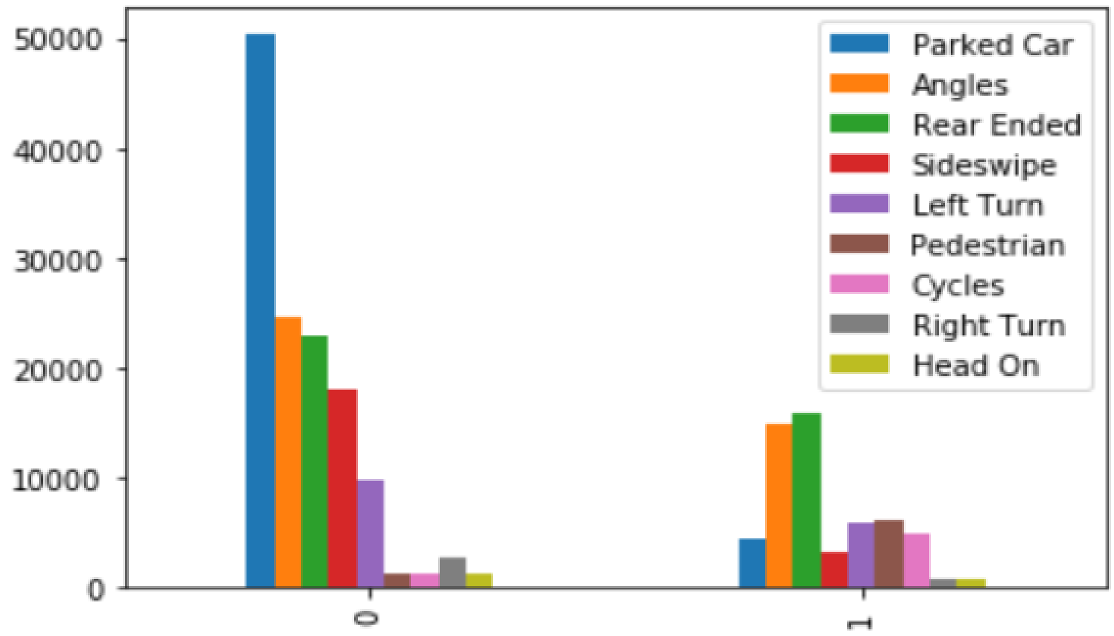




<u>We do the same with all categorical variables but Collision Type.</u>

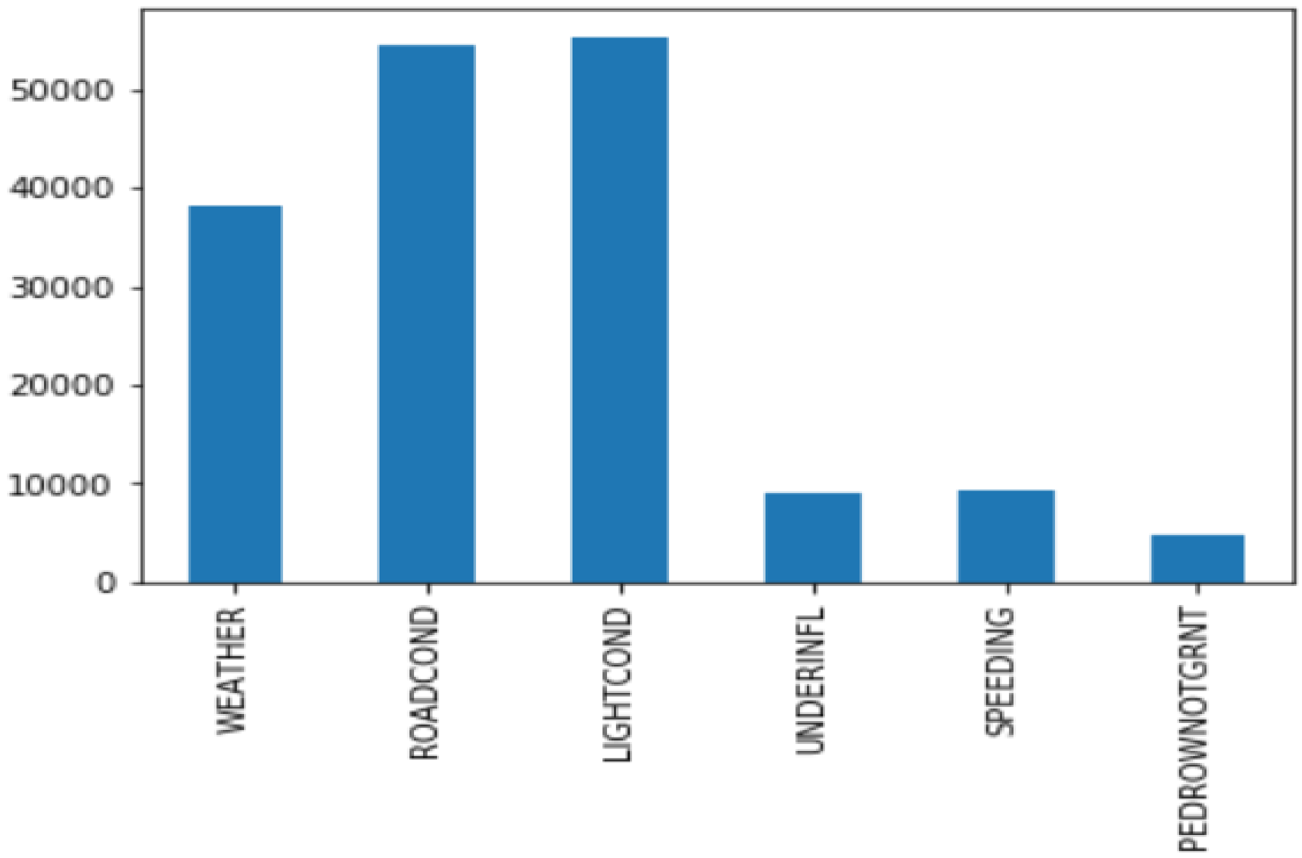
Data Understanding: Predictor Variables (IV)

- Collision Type can be an important factor determining the types of accidents that can lead into a car accident with injures:



Data Understanding: Predictor Variables (V)

- Frequency of the factors that can lead into an accident:



Data Preparation: Data unbalance

- Data in unbalanced ! The target variable has two times more property damage accidents (0) than injure accidents (1). Two options to balanced it :
  - We erase randomly half of the prop. damage accidents (y= 0)
    - Problem: We are loosing a lot of data
  - We insert synthetically the double of values of injure accidents (y = 1)
    - Problem: We are generating a bias and over-valuating the under-represented class and increasing the computational resources.
  - We choose the second option ! We use a library called SMOTE()

```
sum(y_unbalanced)/len(y_unbalanced)
]:  0.30117130076031806

X_bal, y_bal = SMOTE().fit_resample(X_unbalanced, y_unbalanced)

sum(y_bal)/len(y_bal)
]:  0.5
```

Modeling: Overview

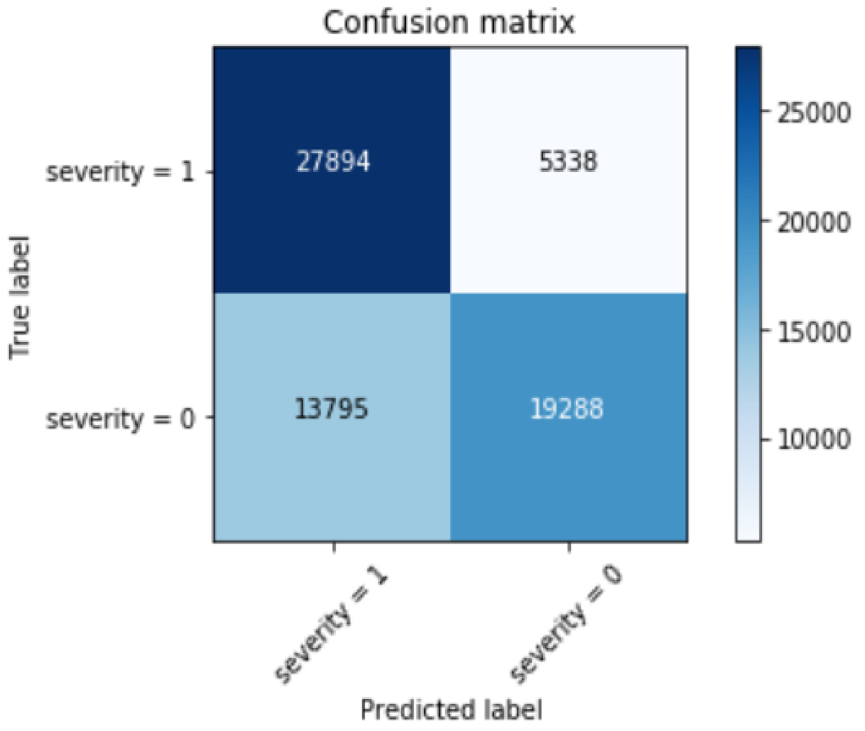
- ❑Data is normalized

- ❑Data is split into two groups:
  - ❑70% train set
  - ❑30% test set

- ❑Machine Learning Models used:
  - ❑Decision Tree Analysis
  - ❑Logistic Regression
  - ❑K-Nearest Neighbor

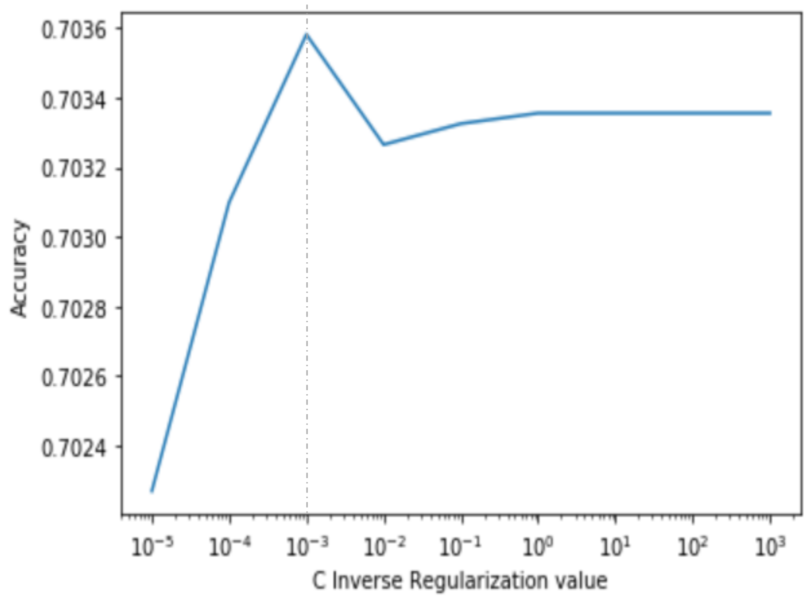
Modeling: Decision Tree Analysis

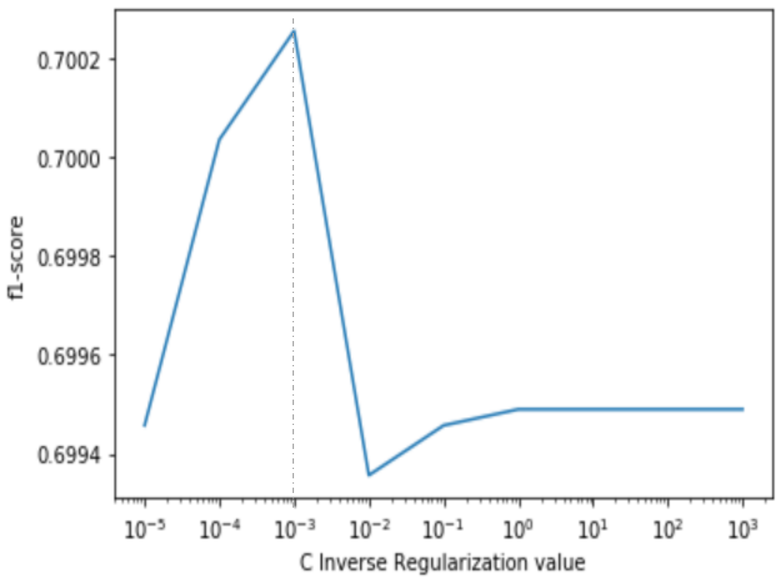
- ❑Manually tuning.
- ❑Max depth = 10
- ❑Both accuracy and f1-score around 0.70

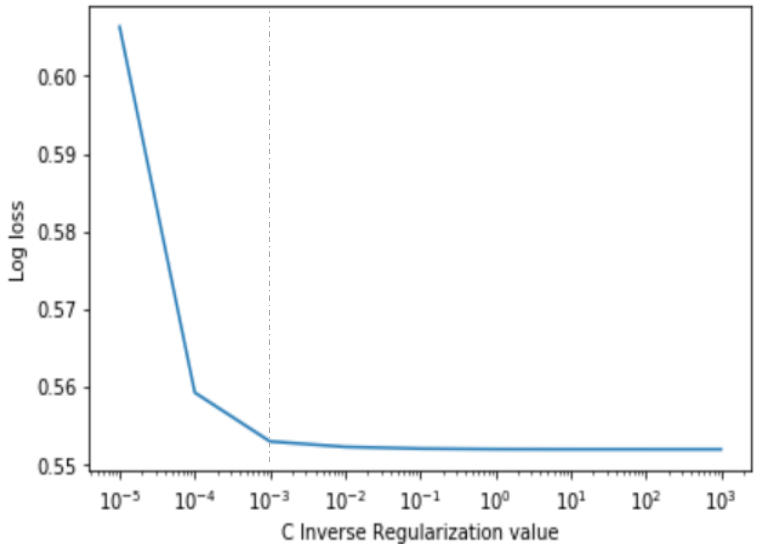


Modeling: Logistic Regression

- Tuning: Regularization inverse constant c.
- Optimal $c \approx 0.001$

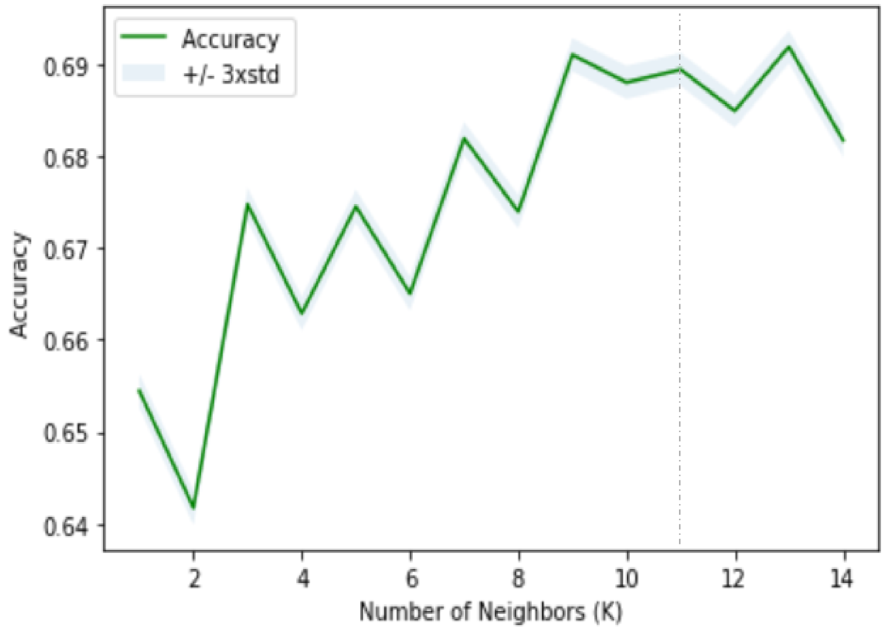


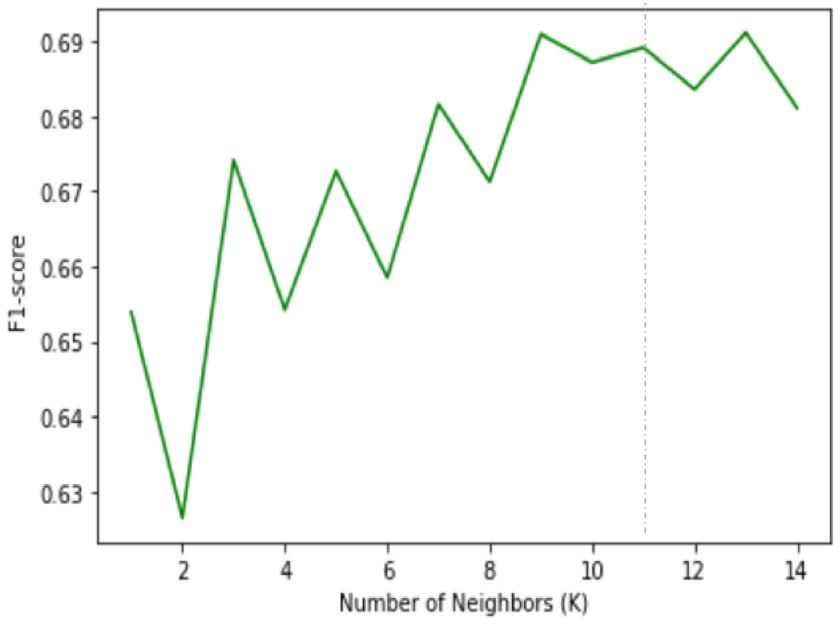




Modeling: k-Nearest Neighbors

- ❑Optimal number neighbours $n \approx 9$.
- ❑Both accuracy and f1-score around 0.66-0.9
- ❑Very slow method !





Evaluation and results

	Tree Decision Analysis			Logistic Regression			K-Nearest Neighbors		
Severity Code	Precision	Recall	F1-score	Precision	Recall	F1-score	Precision	Recall	F1-score
0	0.78	0.58	0.67	0.76	0.60	0.67	0.69	0.69	0.69
1	0.67	0.84	0.74	0.67	0.81	0.73	0.69	0.69	0.69
Weighted	0.73	0.71	0.71	0.71	0.70	0.70	0.69	0.69	0.69
Accuracy	0.71			0.70			0.69		

Best model :
Tree Decision Analysis

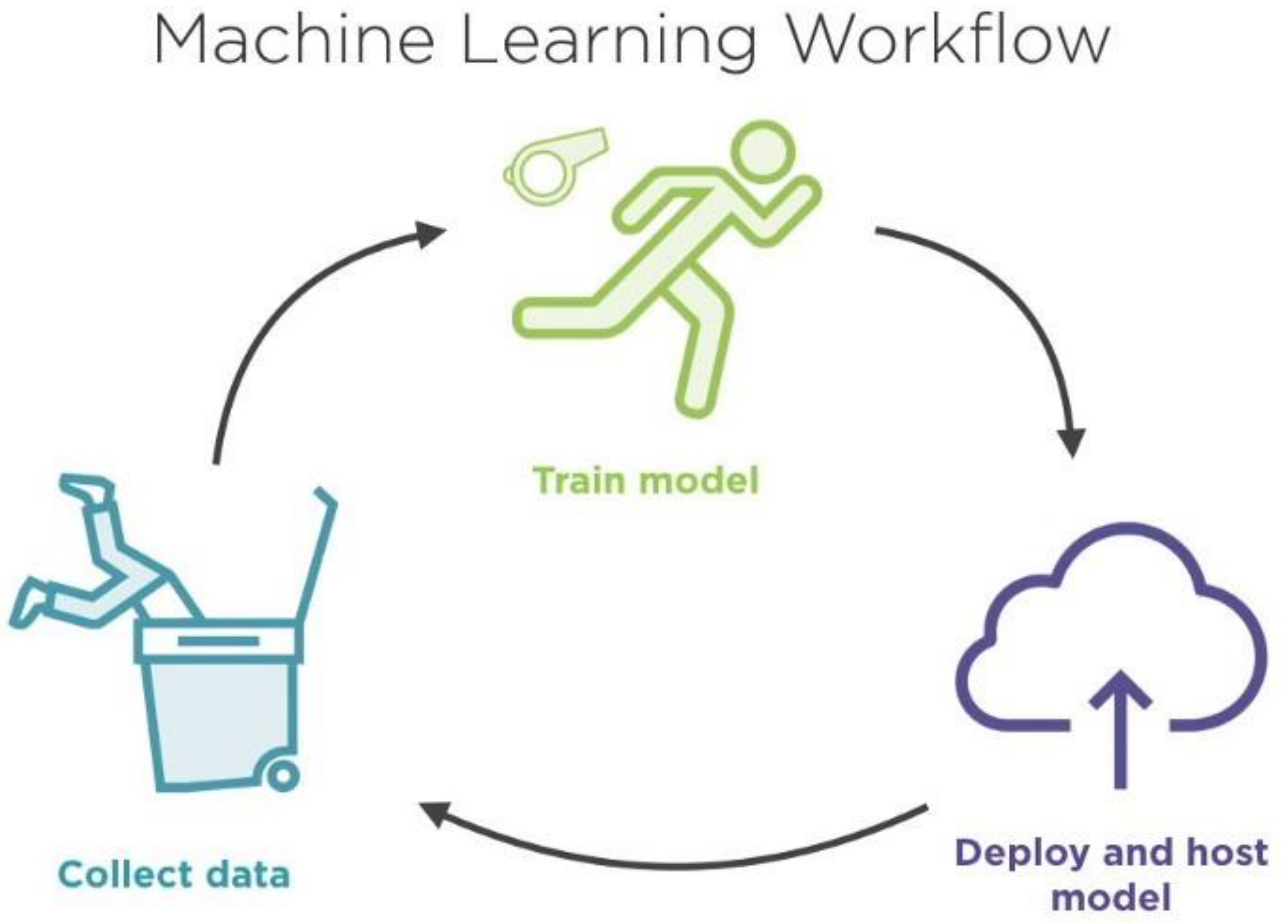
Coeficients values (LR):

```
LR.coef_
]: array([[-0.  , -0.02, -0.05,  0.15,  0.12,  0.05,  0.11,  0.26,  0.75,
         0.62,  0.09,  0.04, -1.97, -1.07, -0.52, -1.33, -2.72, -1.17,
        -1.89, -0.69, -1.72]])
```

The weather, road and light conditions are not a good estimator for knowing the severity of an accident.

At least with our DATA !

Questions?



Machine Learning Workflow
Train model
Deploy and host model
Collect data