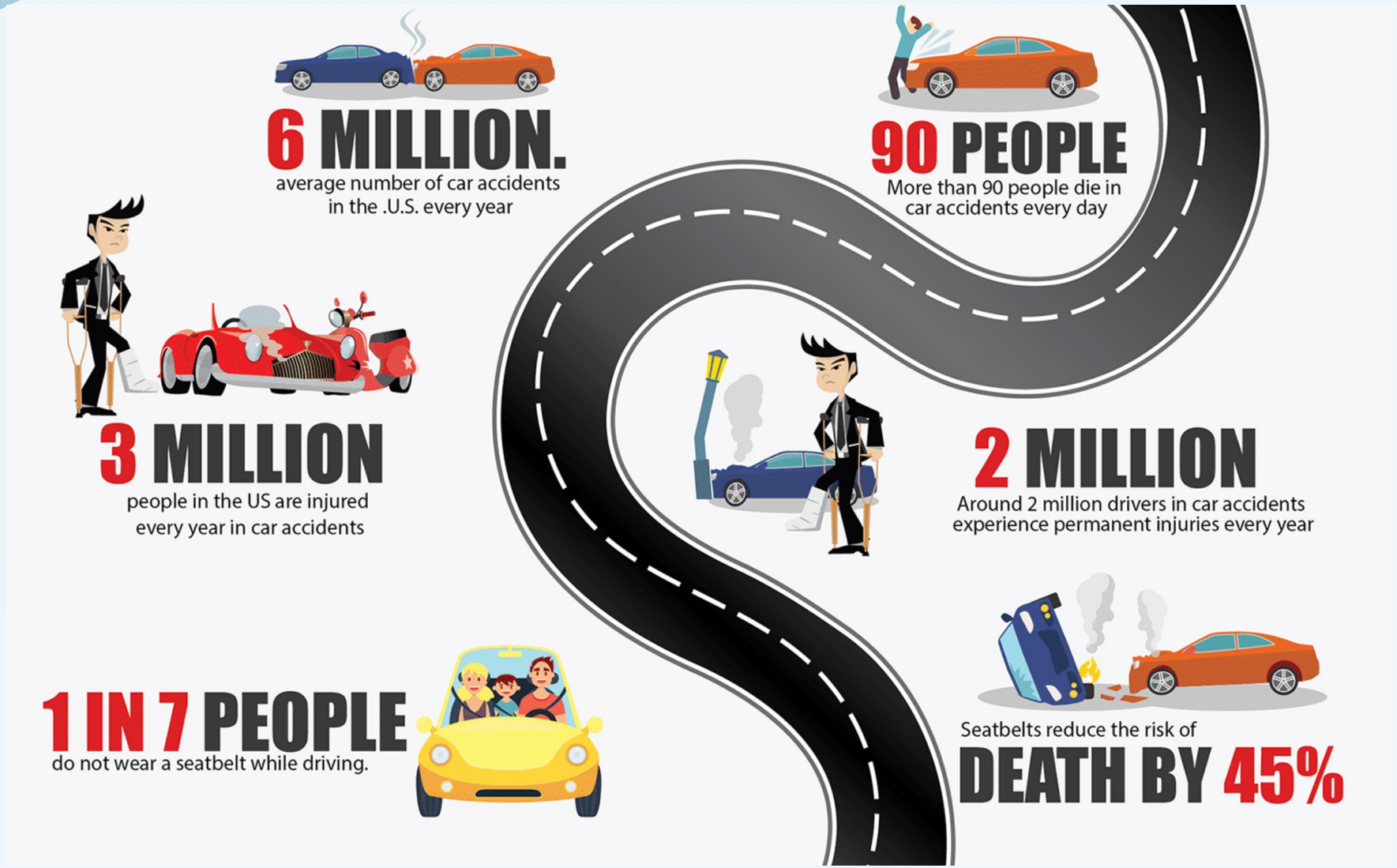
Seattle Accidents Analysis

CONTENTS

01

Introduction

01 Background



01 Background

How to improve road safety?

- Machine learning to predict the likelihood of severe traffic accidents, thereby warning drivers of the dangers
- Predicted severity of accidents could help medical facilities prepare in advance so as to decrease fatalities

Better Awareness

Fewer Fatalities

Less Work for Police

02

Data

02 Data

	SEVERITYCODE	X	Y	OBJECTID	INCKEY	COLDETKEY	REPORTNO	STATUS
0	2	-122.323148	47.703140	1	1307	1307	3502005	Matched
1	1	-122.347294	47.647172	2	52200	52200	2607959	Matched
2	1	-122.334540	47.607871	3	26700	26700	1482393	Matched
3	1	-122.334803	47.604803	4	1144	1144	3503937	Matched
4	2	-122.306426	47.545739	5	17700	17700	1807429	Matched

01 Data Source

Seattle Department of Transportation (SDOT). Updated weekly, from 2004 to present.
Email: DOT_IT_GIS@seattle.gov

02 Metadata

The raw dataset contains 38 columns and 194673 row. Except the first column being the label, all other 37 columns are features.
Complete metadata: click here.

03

Methodology

01 Methodology—Data Preprocessing

Eliminating Bias

- Raw data contains far more instances of SEVERITYCODE 1 than of 2 (around 2.34 : 1)
- Uses dataframe.sample() method to sample from SEVERITYCODE==1 instances an amount equal to the number of SEVERITYCODE==2 instances

01 Methodology—Exploratory Data Analysis

Which features affect the SEVERITYCODE?

- Dataframe.groupby(feature_name)[].value_counts() is used on each column to determine the ones correlated with accident severity
- Converts INCDATE to data objects and then to day of the week, but finds no correlation with SEVERITYCODE

Which Features?

Why these features?

To be one-hot encoded?

01 Methodology—One Hot Encoding

How could categorical features be used to train the model?

- Dataframe[feature_name].replace() was used on each feature to convert categorical variables into numerical ones
- Tests the post-processing dataset with dataframe.dtypes to double check

Numerical Values

Ready for training

Decreased complexity

01 Methodology—Feature Selection and Normalization

How could features on different scales be used without bias?

- Selects 14 features from the dataset, including weather, road condition, lighting, etc.
- Uses dataframe.dropna() to drop rows of the feature set with NaN values and preprocessing.StandardScalar().fit().transform() to normalize the feature set

No empty cells

Without Bias

Ready for training

01 Methodology—Model Training and Testing

How to train the ML models with existing data and test them?

- Uses the train_test_split() method to split the datasets into X_train, y_train, X_test, y_test
- Imports four ML classification models (KNN, Decision Tree, SVM, and Logistic Regression), trains them with X_train and y_train, and tests them with X_test and y_test to obtain their performance

Models trained

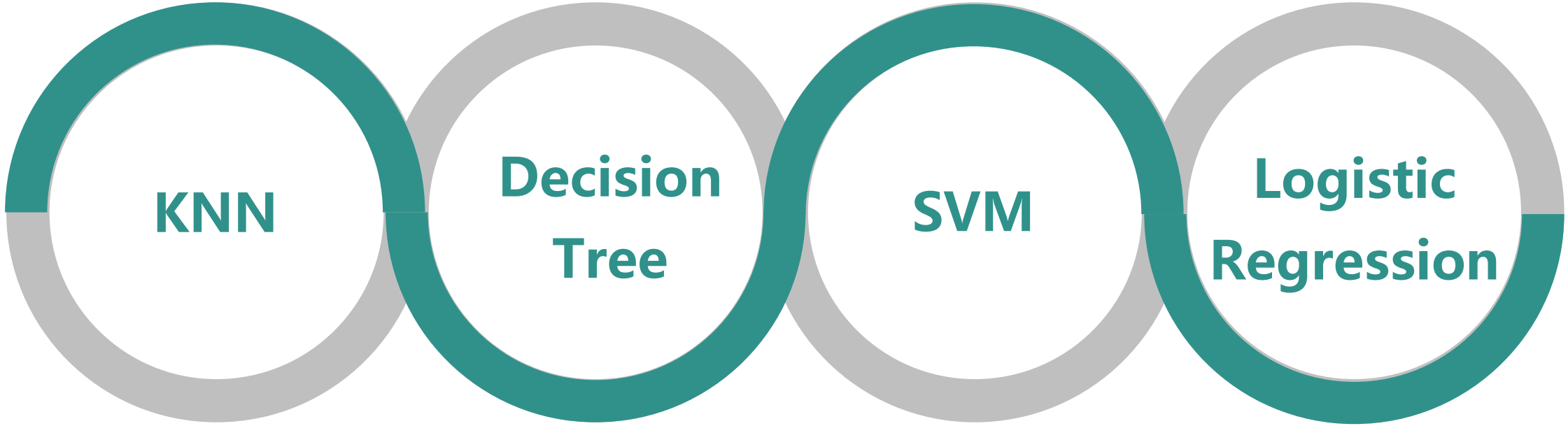
Models tested

Performance Obtained

04

Results & Discussions

03 Results and Discussions



Accuracy Score: 0.694

K=25, consumes the most computing time

0.702

max_depth=15, consumes little computing time

0.692

Kernel = ‘rbf’ , took much computing time

0.659

C=0.1, took moderate computing time

03 Results and Discussions

Model Selection

With the least computing time and the most accurate results, decision tree will be selected for deployment

Lesson Learned

Preparing data, rather than training the models, takes the most time

Improvements

Fine tune the parameters of the ML models so that better results could be predicted

Deployment

After the model is deployed, it should be continually updated with newly-generated data for better performance