AI Boot Camp Project 3

Project 3 Final

Team Members:

- Thomas Brown
- Chris Covill
- Ava Lee
- Jed Murphy

Project Overview

Project Purpose / Description

Analyze a problem using machine learning (ML) or neural network:

We chose to analyze Colorado Traffic Data from Kaggle using a neural network.

Project Goals

Goal/Questions to be addressed

- Goals:

o Determine the best model to analyze all 90,885 Colorado crashes that were collected from February 2016 to March 2023.
o Analyze all Colorado traffic crash data, and record accuracy calculations for Weather, County and Street.

The Data

Data Sources

- **Our Data Source used was "US Accidents: A Countrywide Traffic Accident Dataset"**
- **We used Colorado Accidents only**

Clean and Consistent

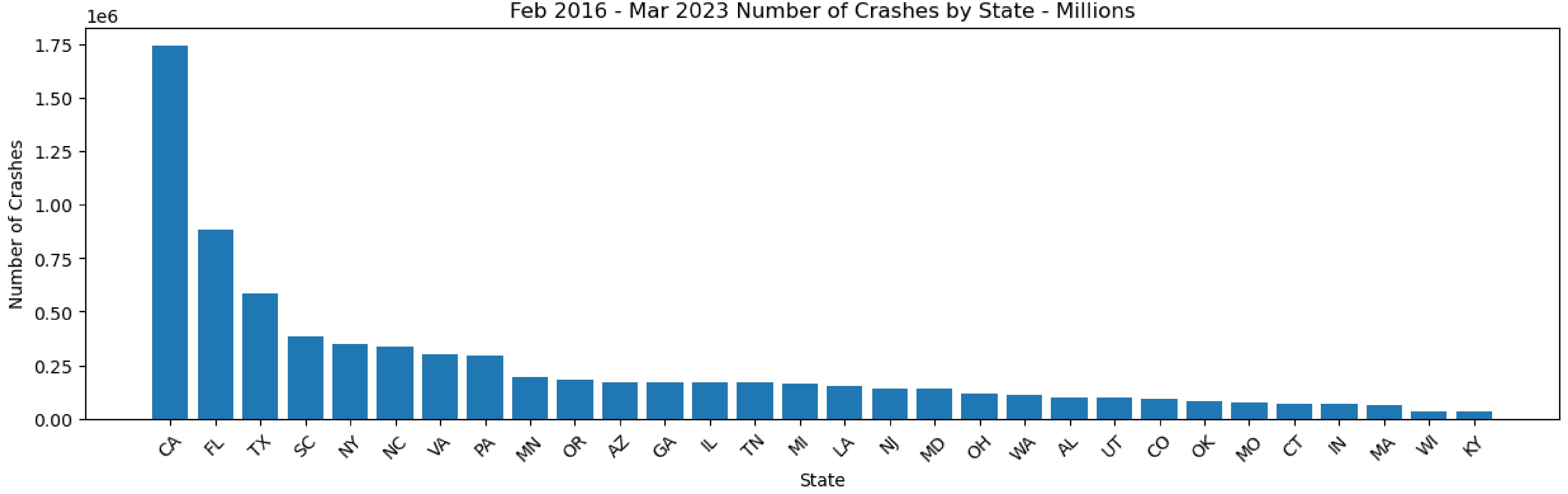
Contains Colorado only
Update County to numeric
Calculate P values for streets
Encode top 30 streets by accidents

Inconsistent and Duplicative

Remove duplicates from Weather Conditions

The Data

Data Sources



Approach

Approach taken to achieve goals

The analysis was broken into steps with a Jupiter notebook for each step:

- Step1 Build base data
- Step2 Analyze weather and county data
- Step3 Analyze street and county data

Performance was measured.

- Best Model: Scikit-learn, Keras, TensorFlow

Performance:

- Step 1 (noPyTorch): 1.83 seconds
- Step 2 (PyTorch): 41.90 seconds
- Step 2 (no PyTorch) 1 minute 51 seconds
- Step 3 (PyTorch): 1 minute 14.06 seconds
- Step 3 (noPyTorch): 3 minute 58 seconds

Project Milestones:

•Project ideation – Complete 7/22
•Data fetching – Complete 7/22
•Data exploration – Complete 7/26
•Data transformation – Complete 7/26
•Data analysis – Complete 7/27
•Testing – Complete 7/27
•Creating documentation – Complete 7/29
•Creating the presentation – Complete 7/30

Approach

Step 2: Street Analysis

Approach taken Analyzing Weather and County Data

The weather analysis:

1. 66 weather categories were combined into 26 (top 10 are shown below)
2. FIPS county codes added.
3. The dense, keras, tensorflow model was compiled with an input and two shared layers and two output layers.
4. Weather Accuracy: .356. County Accuracy: .140

Weather Category	Crashes
Fair	32024
Mostly Cloudy	16447
Partly Cloudy	13865
Cloudy	12382
Light Snow	6424
Snow	1145
Fair _ Windy	1064
Light Rain	1024
Fog	748
Mostly Cloudy _ Windy	574
Total	85697

Approach

Step3: Street Analysis

Approach taken Analyzing Street and County Data

The street analysis was broken into several iterations:

1 Group streets by # accidents to identify top 30 streets

2 Model top 30 streets with county data (x=severity; y=street, y=county)

3 Obtain p-values for top 30 streets

4 Identified 13 streets with p-value <.05

5 Put each of the 13 streets through our model individually

The following slides depict this effort and outcome…..

Approach

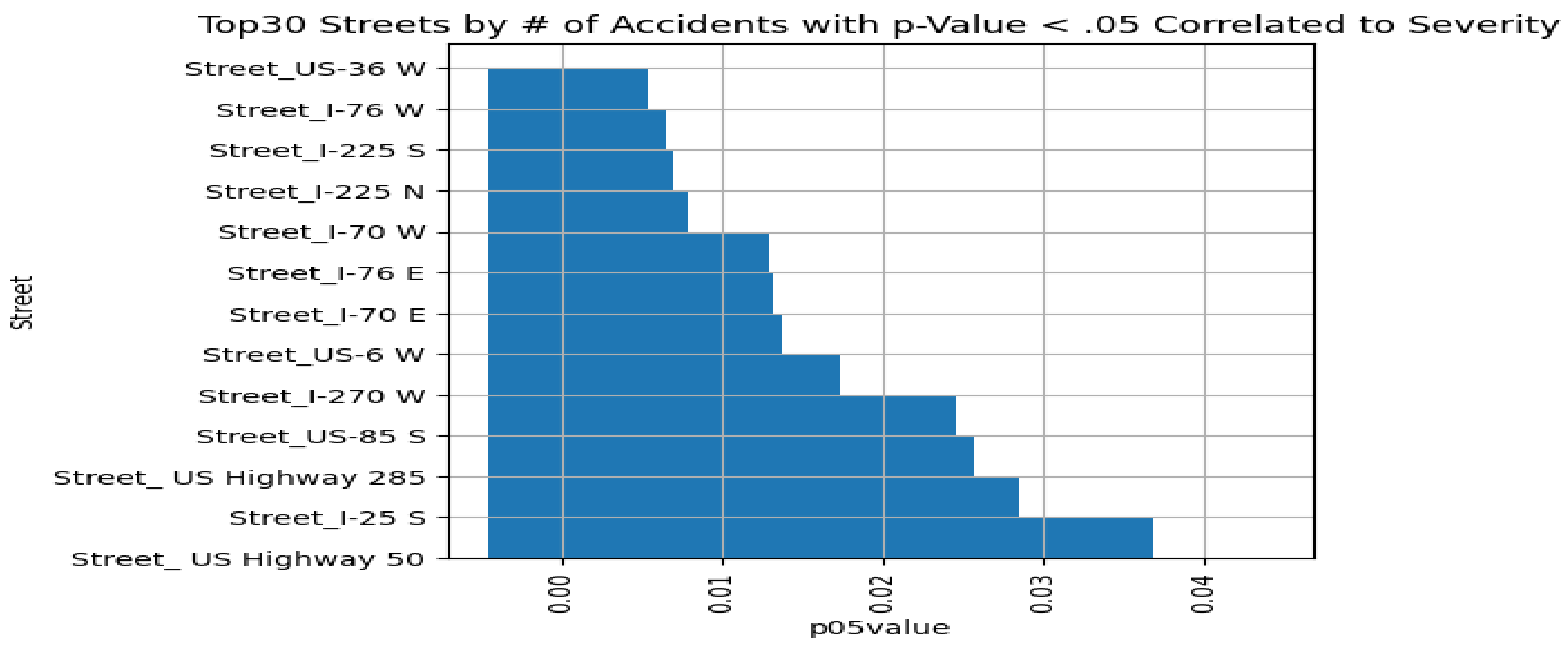
Step3: Street Analysis

Approach taken Analyzing Street and County Data

- o Model top 30 streets with county data – produced low accuracy

  Street Accuracy on top30: 0.11454081535339355
  County Accuracy: 0.059863947331905365

- o Identified 13 streets with p-value <.05 from top 30 streets

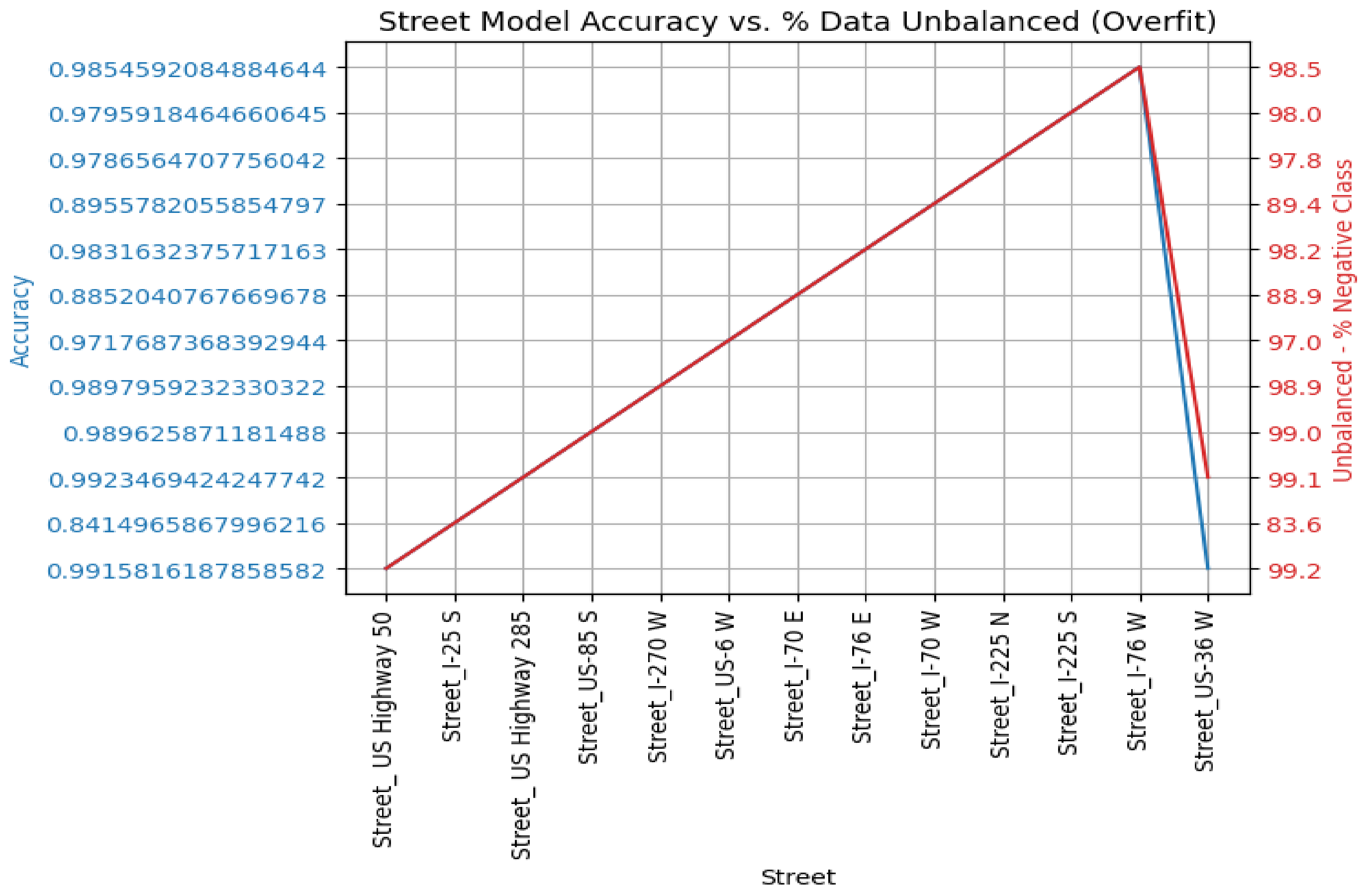


Approach

Step3: Street Analysis

Approach taken Analyzing Street and County Data

Model each of the 13 streets through our model individually



PyTorch

PyTorch Setup:
You must use one additional library or technology NOT covered in class. We chose Windows 11 PyTorch running in Visual Studio with the following steps:
- Install and configure PyTorch on Windows 11 (follow instructions provided by PyTorch).
    - This will enable GUP use by Python.
- Update GPU in Visual Studio:
    - This will enable GPU graphics use wihin Visual Studio
    - Select setting from File-->Preferences-->Settings
    - Search for GPU
    - Set GPU Acceleration = on
    - Check Custom Glyphs
    - Check Enable Images

PyTorch Test Run produced the following results based on the following code:

#GPU
```
b = torch.ones(4000,4000).cuda() # Create matrix on GPU memory
start_time = timer() ➔for _ in range(1000): ➔ b += b
elapsed_time = timer() - start_time ➔print('GPU time = ',elapsed_time)
```
#CPU
```
a = torch.ones(4000,4000) # Create matrix on CPU memory
start_time = timer()➔for _ in range(1000):➔ a += a
elapsed_time = timer() - start_time
```
Results:
```
    GPU time =  0.007673599990084767 seconds
    CPU time =  2.526412000064738 seconds
```

Result/Conclusion

1. The quality the data from Kaggle was inconsistent, causing many hours of rework and modeling for accident prediction.
2. Predicting accidents base on Weather Conditions in our model gave us a 35% accuracy when including all Colorado Accidents.
3. Predicting accidents base on County in our model gave us 14% accuracy when including all Colorado Accidents.
4. Predicting accidents for top 30 streets by accidents, in our model, produced 11% accuracy
5. Predicting accidents for 13 streets out of the top 30 that had P values < .05, produced accuracy ranging from 84% to 99%. However, all 13 showed being overfit. Further analysis to reduce the majority class of negative encoding to produce improved accuracy, is required.

Summary and Future Considerations

Using Colorado Traffic Accident Data from Kaggle only produces mixed accuracy results.

Additional data sets could be used to give a complete picture of all traffic (accidents or not) to improve model accuracy.

The team tried many model configurations the results presented here are representative of the best models created.

We may be able to leverage other data in the Accidents table (Latitude and Longitude) to help improve accuracy.