Clare Robbins, Jordan Holley Riggs, Matt Riley, Sari Broudy, Savannah Posner

Why Tennessee Tornadoes

The members of this team live or have lived in the state of Tennessee. We have noticed a change in the frequency of tornadoes in the state during our collective time living in the state. The team was curious to see if data supports our concerning hypotheses. We believe this information can be used for homebuyers and insurance companies.

The data was obtained from data.world and represents tornado tracks from the United States, Puerto Rico, and the US Virgin Islands. For this project we filtered the data for tornadoes in the state of Tennessee.

What will the data show?

Our Research questions for the data to answer are:

1. Have tornadoes increased in intensity in the last 50 years in the state of Tennessee?
2. What counties are most likely to have more tornadoes?
3. Has the frequency of tornadoes in Tennessee increased since 1950?

Data Exploration

After deciding which data to use, we narrowed our data to just the state of Tennessee using Excel. Then using Python and Pandas to search for missing values, removing columns that were providing information that wasn't needed for our research questions, and to adjust values that switched in 1996 with reporting protocols to match the previous years. Each tornadoes starting and ending counties were calculated in GetCounties.ipynb with the geopy library and exported to counties.csv

Analysis

Machine learning models will be created to predict the following:

1. Magnitude of tornadoes
2. Location of tornadoes
3. Amount of property damage

Tools and Technology

Data Cleaning:

- Python 3.7.13 (pandas and geopy libraries)
- Jupyter Notebook 6.4.8

Database:

- PostgreSQL 11.16
- pgAdmin 4 v6.8
- AWS

Connecting Database:

- Psycopg2

Machine Learning:

- Python (pandas, imbalance-learn, scikit-Learn, numpy libraries)
- Jupyter Notebook

Dashboard:

- Tableau
- Javascript
- Bootstrap
- Leaflet
- D3
- HTML
- CSS

Connecting the Database

Overview of ERD and screenshot of connection stream

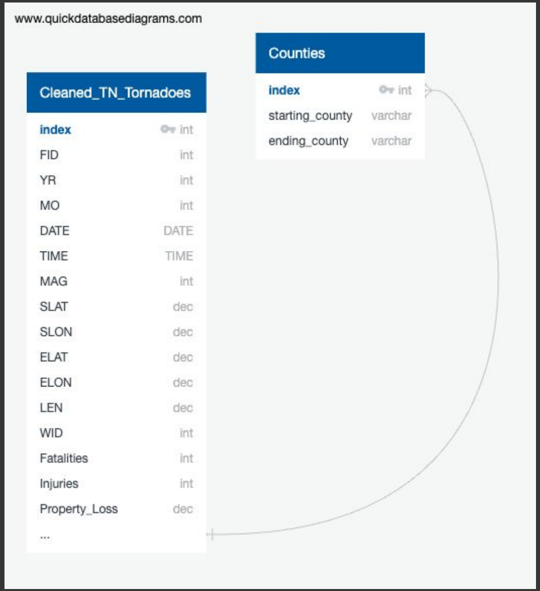
```
try:
    # declare a new PostgreSQL connection object
    conn = connect(
        dbname = "",
        user = "postgres",
        host = "tennesseetornadoes.cdilutmdgtwo.us-east-1.rds.amazonaws.com",
        port = "5432",
        password = password
    )

    # print the connection if successful
    print ("psycopg2 connection:", conn)

except Exception as err:
    print ("psycopg2 connect() ERROR:", err)
    conn = None
```

```
psycopg2 connection:
```

```
cr = conn.cursor()
cr.execute('SELECT * FROM total_tn_tornadoes;')
tmp = cr.fetchall()
```

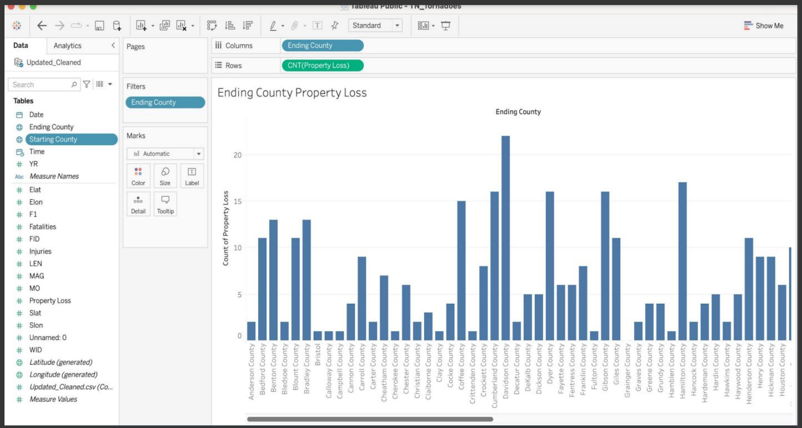
```
# Extract the column names
col_names = []
for elt in cr.description:
    col_names.append(elt[0])
```

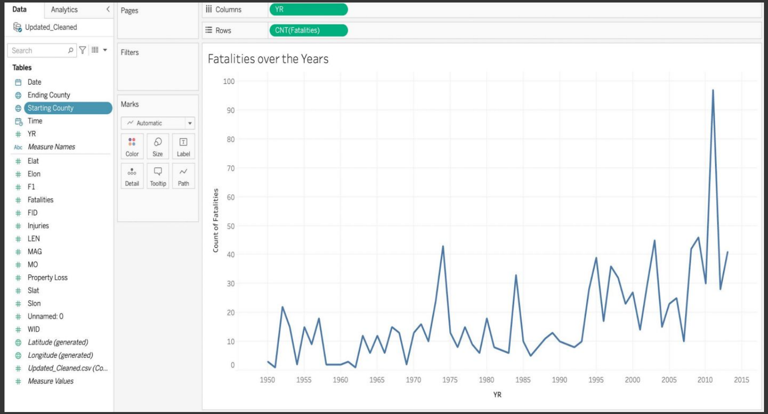
```
# Create the dataframe, passing in the list of col_names extracted from the description
df = pd.DataFrame(tmp, columns=col_names)

df.head()
```

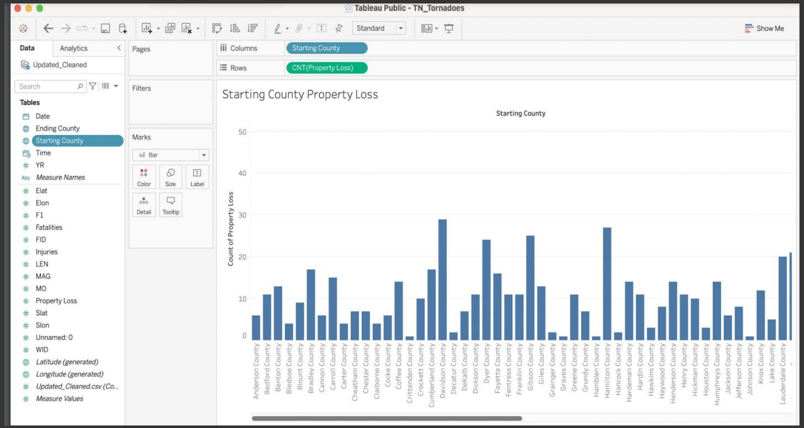


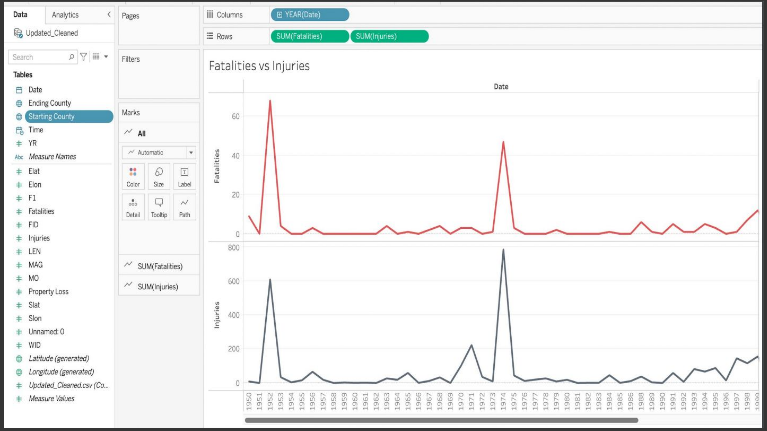
Database

Tables were created in Tableau









Machine Learning:

Machine learning models will be created to predict the following:

1. Magnitude of tornadoes
2. Months in which tornadoes are likely to occur
3. Amount of property damage

We used a trial and error ipynb file to run a variety of models under different circumstances to determine the best model and most notable influencing factors in predicting the respective targets.

We then took that data and transferred it to a final machine learning ipynb file, which displays our final results

Machine Learning

For property loss we were able to achieve 55% accuracy

In [90]:

```python
# prepare the dataframe

df_3 = df.drop(['time',"wid","starting_county",
               'slat','slon','elat','elon','len','ending_county'], axis=1)
# df_3.columns
# df_3['property_loss']=df_3['property_loss'].astype('int')

target = 'property_loss'
X = pd.get_dummies(df_3.drop([target],axis = 1))


# Create our target
y = df[target]


X_train, X_test, y_train, y_test = train_test_split(X,
                                                    y,
                                                    random_state=1)

from sklearn.datasets import fetch_covtype
from sklearn.pipeline import make_pipeline
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import MinMaxScaler
from sklearn.kernel_approximation import PolynomialCountSketch
from sklearn.linear_model import LogisticRegression

pipe = make_pipeline(
    MinMaxScaler(),
    PolynomialCountSketch(degree=2, n_components=300),
    LogisticRegression(max_iter=1000),
)


print(f"Accuracy Score: {round(pipe.fit(X_train, y_train).score(X_test, y_test)*100,2)}%")
```

```
Accuracy Score: 54.07%
```

In [140…

```python
# Create our features

target = 'mag'
X = pd.get_dummies(df_2.drop(columns=target))


# Create our target
y = df[target]


X_train, X_test, y_train, y_test = train_test_split(X,
                                                    y,
                                                    random_state=1)

pipe = make_pipeline(
    StandardScaler(),
    MinMaxScaler(),


    PolynomialCountSketch(degree=6, n_components=850),
    LogisticRegression(solver='sag',max_iter=1000),
)

print(f"Accuracy Score: {round(pipe.fit(X_train, y_train).score(X_test, y_test),3)*100}%")
```

```
Accuracy Score: 55.2%
```

For magnitude we were able to achieve 54% accuracy

Run the second model

target = Month

yields approx 80% accuracy

In [21]:

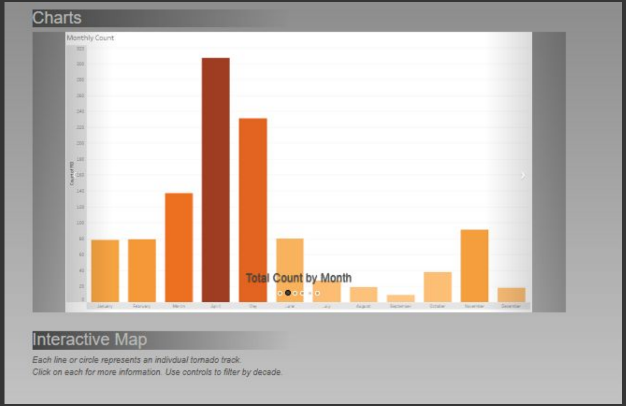
```python
#sklearn.linear_model.SGDClassifier

df_1 = df.drop(['time','starting_county','fatalities','wid',
               'slat','slon','elat','elon','len','ending_county','injuries'], axis=1)

# Create our features

target = 'mo'
X = pd.get_dummies(df_1.drop(columns=target))


# Create our target
y = df[target]


X_train, X_test, y_train, y_test = train_test_split(X,
                                                    y,
                                                    random_state=1)

pipe = make_pipeline(
    MinMaxScaler(),
    PolynomialCountSketch(degree=2, n_components=650),
    LogisticRegression(max_iter=1000),
)

print(f"Accuracy Score: {round(pipe.fit(X_train, y_train).score(X_test, y_test),3)*100}%")
```

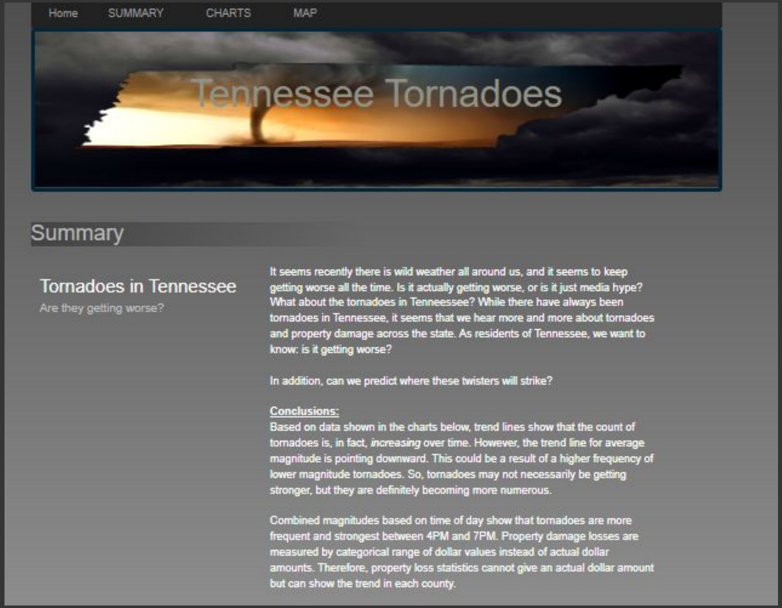
```
Accuracy Score: 80.2%
```

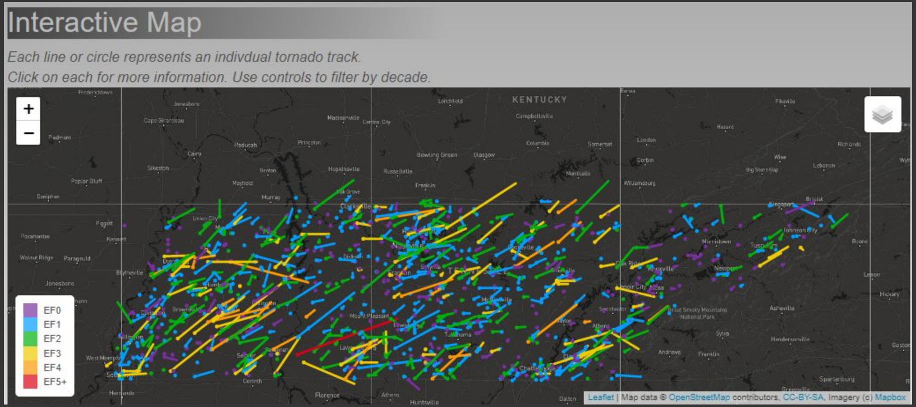
For month of tornadoes we were able to achieve 80% accuracy

Dashboard

The Dashboard has been created using Javascript to be displayed as an interactive webpage.Tableau, CSS, D3,HTML, and Bootstrap components have been used to enhance the displays. The map was created using Leaflet







Result of Analysis:

Based on data, the count of tornadoes is, in fact, *increasing* over time. However, the average magnitude appears to decrease over time. This could be a result of a higher frequency of lower magnitude tornadoes. So, tornadoes may not necessarily be getting stronger, but they are becoming more numerous.

Combined magnitudes based on time of day show that tornadoes are more frequent and strongest between 4PM and 7PM. Property damage losses are measured by categorical range of dollar values instead of actual dollar amounts. Therefore, property loss statistics cannot give an actual dollar amount but can show the trend in each county.

With 80% accuracy we are able to predict the months in which tornadoes are likely to occur.

Recommendations for Future Analysis: