CALIFORNIA WEATHER & WILDFIRES

Breanna Sewell

PROJECT MOTIVATION & SUMMARY

This analysis aims to determine if there are correlations between meteorological factors and wildfire extent or duration within California over the span of 8 years

- Wildfire variables analyzed:
  - Wildfire extent – acres burned
  - Wildfire duration – number of days fire lasted

- Weather variables analyzed:
  - DX90 – number of days over 90˚F
  - HX01 – extreme maximum soil temperature (˚F)
  - EMXP – extreme maximum daily precipitation (in.)
  - EMXT – extreme maximum daily temperature (˚F)
  - PRCP – total annual precipitation (in.)
  - WSFG – peak wind gust speed (mph)
  - WSF2 – maximum 2 min gust speed (mph)
  - TAVG – average annual temperature (˚F)
  - TMAX – average annual maximum temperature (˚F)

- My findings suggest there is little to no correlation between any of the weather variables and wildfire variables, at least at the scale with which the analysis was conducted

DATA SOURCES

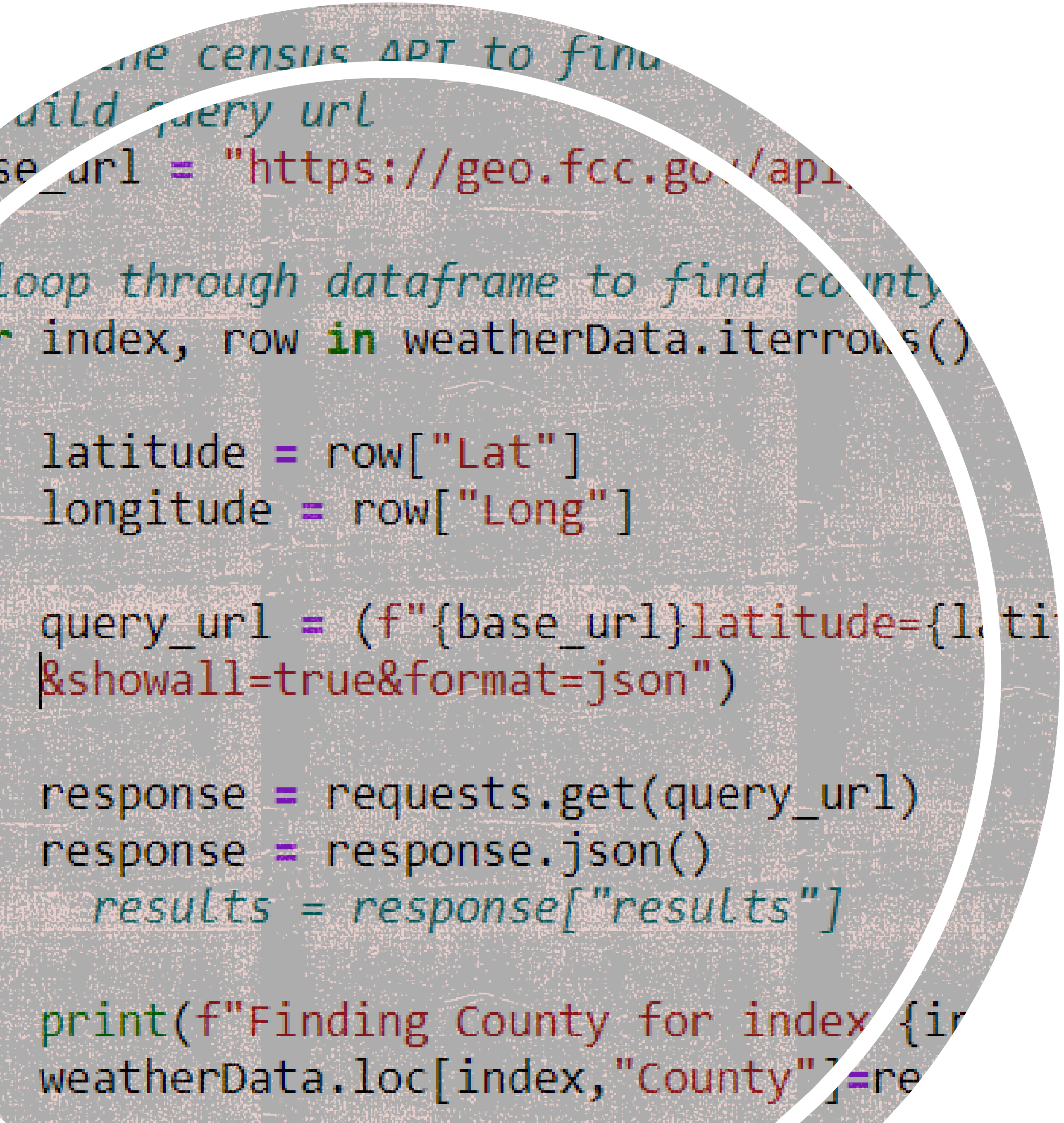
Wildfire Data

- 2013-2020 California wildfires – name, coordinates, start time/date, end time/date, acres, duration
- Source: California Department of Forestry and Fire Protection (CAL FIRE)

Weather Data

- 2013-2020 California weather station records – station, coordinates, year, DX90, HX01, EMXP, EMXT, PRCP, WSFG, WSF2, TAVG, TMAX
- Source: National Oceanic and Atmospheric Administration (NOAA)

CLEANING


census API to find
query url
url = "https://geo.fcc.go
loop through dataframe to find
index, row in weatherData.iterro
latitude = row["Lat"]
longitude = row["Long"]
query_url = (f"{base_url}latitude={l
&showall=true&format=json")
response = requests.get(query_url)
response = response.json()
results = response["results"]
print(f"Finding County for index {i
weatherData.loc[index,"County"]=re

DATA CLEANING

```
# use the census API to find each entry's county using coordinates
# build query url
base_url = "https://geo.fcc.gov/api/census/block/find?"

# loop through dataframe to find county for each lat long
for index, row in weatherData.iterrows():

    latitude = row["Lat"]
    longitude = row["Long"]

    query_url = (f"{base_url}latitude={latitude}&longitude={longitude}
    &showall=true&format=json")

    response = requests.get(query_url)
    response = response.json()
#     results = response["results"]

    print(f"Finding County for index {index}")
    weatherData.loc[index,"County"]=response["County"]["name"]
```

- Cleaned wildfire and weather data separately using pandas in Jupyter Notebook
- Only common variable between the datasets was coordinates… which was too difficult to merge on
- Used the Federal Communications Commission (FCC) census API to look up county using coordinates for each row of weather data (5,000+ rows)

DATA CLEANING

```
dt32 = weatherData[["Station ID","Station Name","County","Lat","Long","Year","DT32"]]
dx90 = weatherData[["Station ID","Station Name","County","Lat","Long","Year","DX90"]]
emxp = weatherData[["Station ID","Station Name","County","Lat","Long","Year","EMXP"]]
emxt = weatherData[["Station ID","Station Name","County","Lat","Long","Year","EMXT"]]
hx01 = weatherData[["Station ID","Station Name","County","Lat","Long","Year","HX01"]]
prcp = weatherData[["Station ID","Station Name","County","Lat","Long","Year","PRCP"]]
tavg = weatherData[["Station ID","Station Name","County","Lat","Long","Year","TAVG"]]
tmax = weatherData[["Station ID","Station Name","County","Lat","Long","Year","TMAX"]]
tmin = weatherData[["Station ID","Station Name","County","Lat","Long","Year","TMIN"]]
wsf2 = weatherData[["Station ID","Station Name","County","Lat","Long","Year","WSF2"]]
wsfg = weatherData[["Station ID","Station Name","County","Lat","Long","Year","WSFG"]]
```

```
# drop NaNs for each dataframe
dt32 = dt32.dropna()
dx90 = dx90.dropna()
emxp = emxp.dropna()
emxt = emxt.dropna()
prcp = prcp.dropna()
tavg = tavg.dropna()
tmax = tmax.dropna()
tmin = tmin.dropna()

# hx01 only has 7 data points - removing this variable from the analysis
hx01 = hx01.dropna()

# wsfg only has 2 data points - removing this variable from the analysis
wsfg = wsfg.dropna()

# wsf2 only has 540 data points - removing this variable from the analysis
wsf2 = wsf2.dropna()
```

- Had to break weather variables into separate data frames before dropping nulls in order to preserve as much data as possible
- After dropping nulls, 3 of the weather variables had insufficient records for analysis and were removed
  - HX01
  - WSFG
  - WSF2

```python
# prcp - average inches per county
prcp_grouped = prcp.groupby(["Year","County"])
prcp_CountyMean = prcp_grouped["PRCP"].mean()
pd.DataFrame(prcp_CountyMean)

# tavg - average temperature per county
tavg_grouped = tavg.groupby(["Year","County"])
tavg_CountyMean = tavg_grouped["TAVG"].mean()
pd.DataFrame(tavg_CountyMean)

# tmax - average temperature per county
tmax_grouped = tmax.groupby(["Year","County"])
tmax_CountyMean = tmax_grouped["TMAX"].max()
pd.DataFrame(tmax_CountyMean)
```

```python
# DX90 - average days per county
dx90_grouped = dx90.groupby(["Year","County"])
dx90_CountyMean = dx90_grouped["DX90"].mean()
pd.DataFrame(dx90_CountyMean)

# emxp - maximum inches per county
emxp_grouped = emxp.groupby(["Year","County"])
emxp_CountyMax = emxp_grouped["EMXP"].max()
pd.DataFrame(emxp_CountyMax)

# emxt - maximum temperature per county
emxt_grouped = emxt.groupby(["Year","County"])
emxt_CountyMax = emxt_grouped["EMXT"].max()
pd.DataFrame(emxt_CountyMax)
```

DATA CLEANING

- Each weather variable's data frame was then grouped by year and county and aggregated using the most appropriate function (mean or max)
- Each data frame was then exported to csv

DATA CLEANING

```
# create a column that contains the duration of each fire
# first convert the date columns to datetime
fireData["Date Started"] = pd.to_datetime(fireData["Date Started"])
fireData["Date Extinguished"] = pd.to_datetime(fireData["Date Extinguished"])

# subtract the two dates
fireData["Duration (Days)"] = fireData["Date Extinguished"] - fireData["Date Started"]

# convert duration to string and remove "days"
fireData["Duration (Days)"] = fireData["Duration (Days)"].astype(str)
fireData["Duration (Days)"] = fireData["Duration (Days)"].str.replace("days","")

# convert NaT to NaN and convert back to float
fireData["Duration (Days)"] = fireData["Duration (Days)"].replace(["NaT"],"NaN")
fireData["Duration (Days)"] = fireData["Duration (Days)"].astype(float)

# create a column that holds the year of each start date
fireData["Year"] = fireData["Date Started"].dt.year
```

- To get the duration of each wildfire, the date started and date extinguished were converted to ‘datetime’
- The date started column was subtracted from the date extinguished column to get duration in days
- Year of start date was extracted to its own column

DATA CLEANING

```
# separate into two dataframes
fireDamage = fireData[["Name","Year","County",
                       "Lat","Long","Acres Burned"]]
fireDuration = fireData[["Name","Year","County",
                       "Lat","Long","Duration (Days)"]]

# remove any NaNs from each dataframe
fireDamage = fireDamage.dropna()
fireDuration = fireDuration.dropna()
```

```
# groupby year and county and sum for each variable
fireDamageCounty = fireDamage.groupby(["Year","County"])
fireDamageCounty = fireDamageCounty["Acres Burned"].sum()
pd.DataFrame(fireDamageCounty)

fireDurationCounty = fireDuration.groupby(["Year","County"])
fireDurationCounty = fireDurationCounty["Duration (Days)"].sum()
pd.DataFrame(fireDurationCounty)
```

- All NaT's were converted to NaNs
- Like the weather data, the two wildfire variables were broken into separate data frames before the nulls were dropped in order to preserve data
- Each wildfire variable's data frame was then grouped by year and county and aggregated by sum
- Each data frame was then exported to csv

ANALYSIS

```
... variables by year
...geYearTotals ... groupby(["Y...
...amageYe...otals = damageYearTotals["Acres Bu...

d...ationYearTotals = fireDuration.groupby([...ear"]...
durationYearTotals = durationYearTotals["Dura...on (D...

dx90YearTotals = dx90.groupby(["Year"])
dx90YearTotals = dx90YearTotals["DX90"].sum()

emxpYearTotals = emxp.groupby(["Year"])
emxpYearTotals = emxpYearTotals["EMXP"].sum()

emxtYearTotals = emxt.groupby(["Year"])
emxtYearTotals = emxtYearTotals["EMXT"].sum()

prcpYearTotals = prcp.groupby(["Year"])
prcpAveYearTotals = prcpYearTotals["PRCP"].mean()
prcpSumYearTotals = prcpYearTotals["PRCP"].sum()

tavgYearTotals = tavg.groupby(["Year"])
tavgYearTotals = tavgYearTotals["TAVG"].mean()

tmaxYearTotals = tmax.groupby(["Year"])
tmaxAveYearTotals = tmaxYearTotals["TMAX"].mean()
tmaxSumYearTotals = tmaxYearTotals["TMAX"].sum()

tminYearTotals = tmin.groupby(["Year"])
tminAveYearTotals = tminYearTotals["TMIN"].mean()
tminSumYearTotals = tminYearTotals["TMIN"].sum()
```

DATA ANALYSIS

```python
# merge DX90 and fire extent
damageDX90 = pd.merge(fireDamage, dx90, on=["Year","County"])
damageDX90
```

	Year	County	Acres Burned	DX90
0	2013	Alameda	328.0	19.900000
1	2013	Amador	96.0	45.500000
2	2013	Butte	3237.0	63.625000
3	2013	Calaveras	77.0	62.000000
4	2013	Contra Costa	3877.0	30.375000
...	...	...	...	...
345	2020	Trinity	117.0	55.333333
346	2020	Tulare	1397.0	52.583333
347	2020	Tuolumne	2867.0	83.250000
348	2020	Ventura	4266.0	61.625000
349	2020	Yuba	2467.0	69.500000

350 rows × 4 columns

- The intent was to assess correlation between each wildfire variable and each weather variable at the county level
- After merging the first two data frames, wildfire extent and DX90, many rows dropped
- This was because many fires span multiple counties, which hadn't occurred to me previously
- This is problematic not only because of the loss of data, but because of the type of data lost – generally speaking, fires that span multiple counties are the largest fires – which are critical data points

DATA ANALYSIS

```
# group all variables by year
damageYearTotals = fireDamage.groupby(["Year"])
damageYearTotals = damageYearTotals["Acres Burned"].sum()

durationYearTotals = fireDuration.groupby(["Year"])
durationYearTotals = durationYearTotals["Duration (Days)"].sum()

dx90YearTotals = dx90.groupby(["Year"])
dx90YearTotals = dx90YearTotals["DX90"].sum()

emxpYearTotals = emxp.groupby(["Year"])
emxpYearTotals = emxpYearTotals["EMXP"].sum()

emxtYearTotals = emxt.groupby(["Year"])
emxtYearTotals = emxtYearTotals["EMXT"].sum()

prcpYearTotals = prcp.groupby(["Year"])
prcpAveYearTotals = prcpYearTotals["PRCP"].mean()
prcpSumYearTotals = prcpYearTotals["PRCP"].sum()

tavgYearTotals = tavg.groupby(["Year"])
tavgYearTotals = tavgYearTotals["TAVG"].mean()

tmaxYearTotals = tmax.groupby(["Year"])
tmaxAveYearTotals = tmaxYearTotals["TMAX"].mean()
tmaxSumYearTotals = tmaxYearTotals["TMAX"].sum()

tminYearTotals = tmin.groupby(["Year"])
tminAveYearTotals = tminYearTotals["TMIN"].mean()
tminSumYearTotals = tminYearTotals["TMIN"].sum()
```

- Instead, the analysis had to be done at a higher level
- Each of the data frames (both weather and wildfire variables) were grouped by year and aggregated using the most appropriate function per variable (e.g., sum, mean, max)
- Some weather variables were aggregated using multiple functions

Year	Acres Burned	Duration (Days)	DX90	EMXP	EMXT	PRCP Ave	PRCP Sum	TAVG	TMAX Ave	TMAX Sum	TMIN Ave	TMIN Sum
2013	496123.0	452.0	2708.413460	79.33	5560.0	8.717269	435.863464	57.975776	70.732775	3607.371516	45.203894	2305.398585
2014	297186.0	825.0	3449.543725	225.33	6029.0	24.397685	1390.668038	60.389203	72.482017	4058.992963	48.346128	2707.383143
2015	332622.0	17185.0	3202.943617	179.55	6153.0	16.394027	950.853586	59.851378	71.826107	4094.088079	47.897763	2730.172464
2016	452406.0	18430.0	3103.878645	195.53	6120.0	30.553554	1772.106130	58.880800	70.492438	4018.068973	47.316859	2697.060968
2017	1264155.0	70265.0	3432.177368	226.89	6242.0	34.946927	2026.921759	59.184024	70.768419	3963.031492	47.611580	2666.248492
2018	1531391.0	50930.0	2994.784890	181.03	6040.0	21.840896	1266.771962	58.904042	70.835184	3966.770314	46.989238	2631.397326
2019	285439.0	998.0	2689.396306	225.04	5953.0	34.196570	1983.401056	57.791823	68.999768	3863.986981	46.577024	2608.313362
2020	2521233.0	1579.0	3725.257453	82.04	6227.0	11.509818	575.490908	59.744531	71.824985	4022.199176	47.693593	2670.841197

DATA ANALYSIS

- The variables were then added back to a common data frame
- Each combination of variables was plotted on a scatterplot and the r-value was calculated

RESULTS

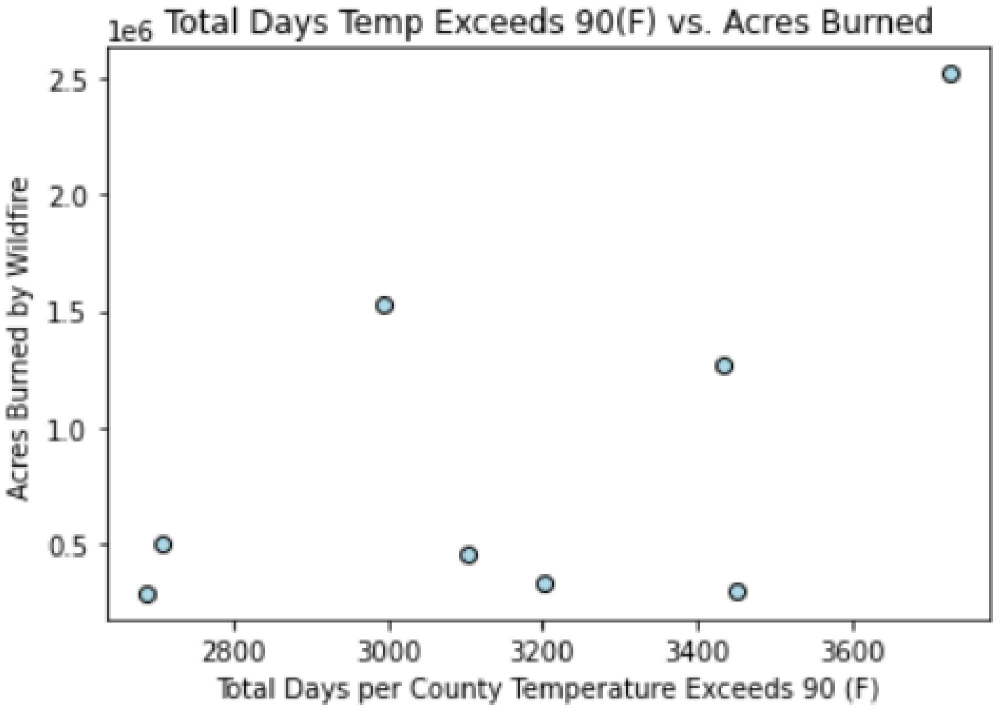
```
DX90 and fire extent
ageDX90 = pd.merge(fireDamage, dx
damageDX90
```

Year	County	Acres Burned	DX90
2013	Alameda	328.0	19.900000
2013	Amador	96.0	45.500000
2013	Butte	3237.0	63.625000
2013	Calaveras	77.0	62.000000
2013	Contra Costa	3877.0	30.375000
...	...	...	...
2020	Trinity	117.0	55.333333
2020	Tulare	1397.0	52.583333
2020	Tuolumne	2867.0	83.250000
2020	Ventura	4266.0	61.625000
2020	Yuba	2467.0	69.500000

ows × 4 columns

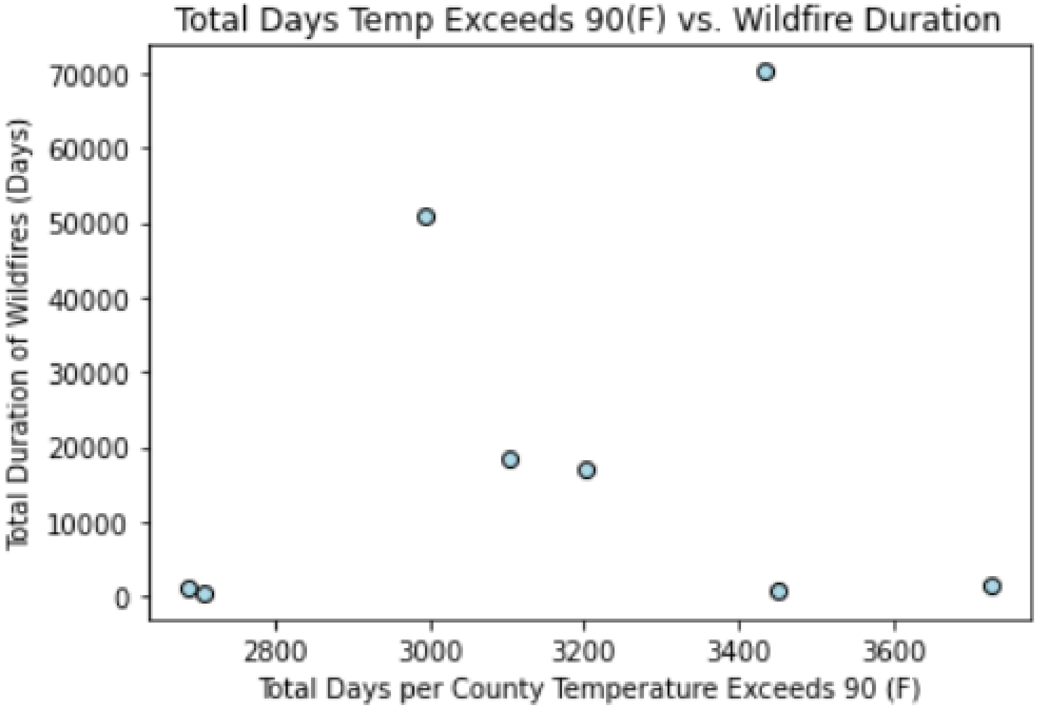
DX90 vs. Fire Extent

r-value: 0.58, moderate correlation



DX90 vs. Fire Duration

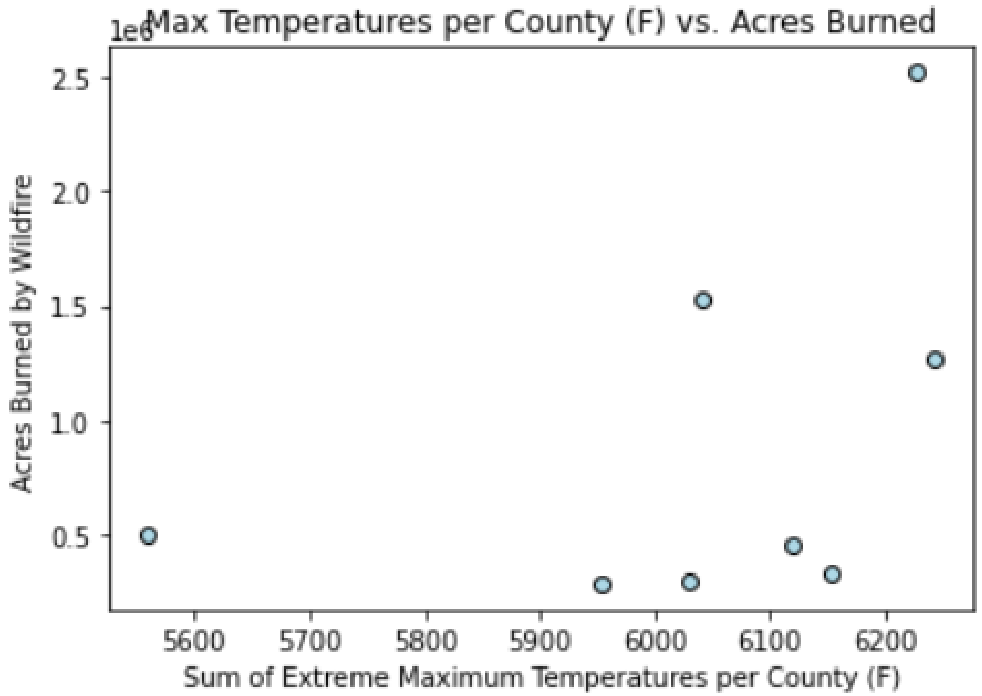
r-value: 0.15, no correlation



RESULTS: DX90

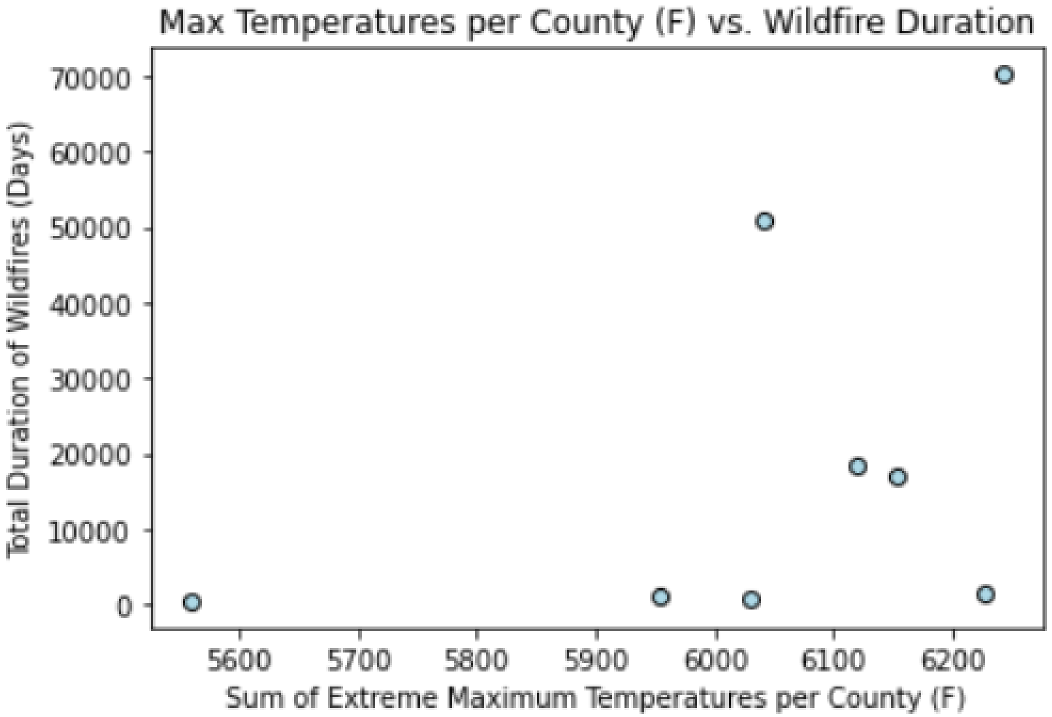
EMXT vs. Fire Extent

r-value: 0.43, weak correlation



EMXT vs. Fire Duration

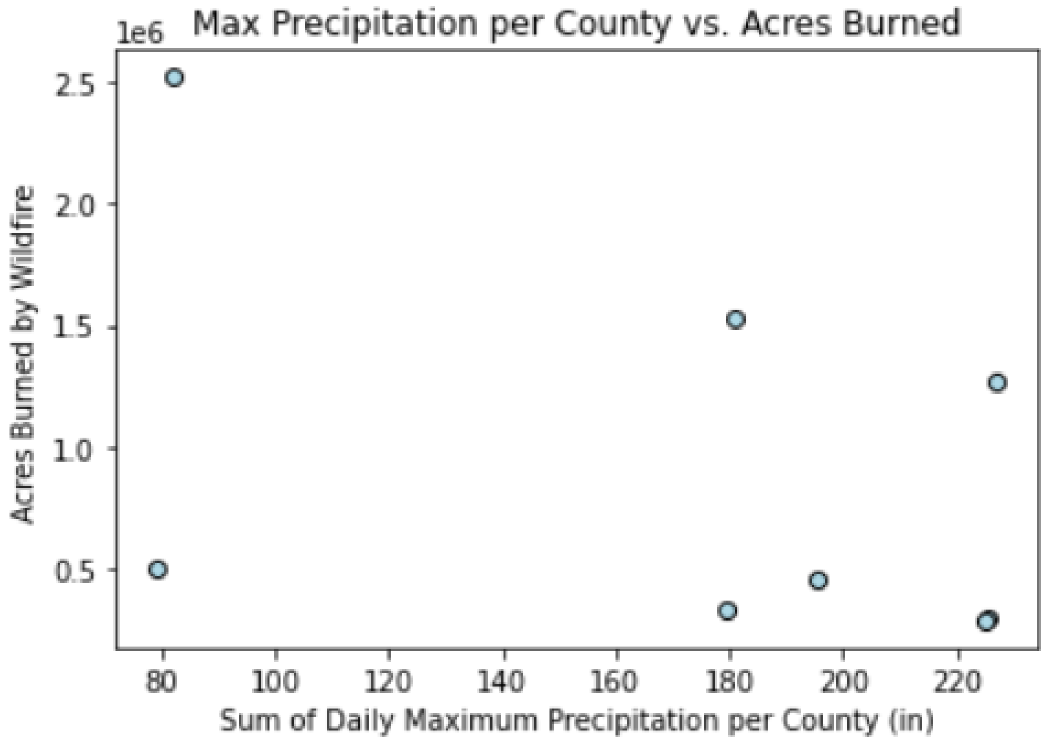
r-value: 0.43, weak correlation



RESULTS: EMXT

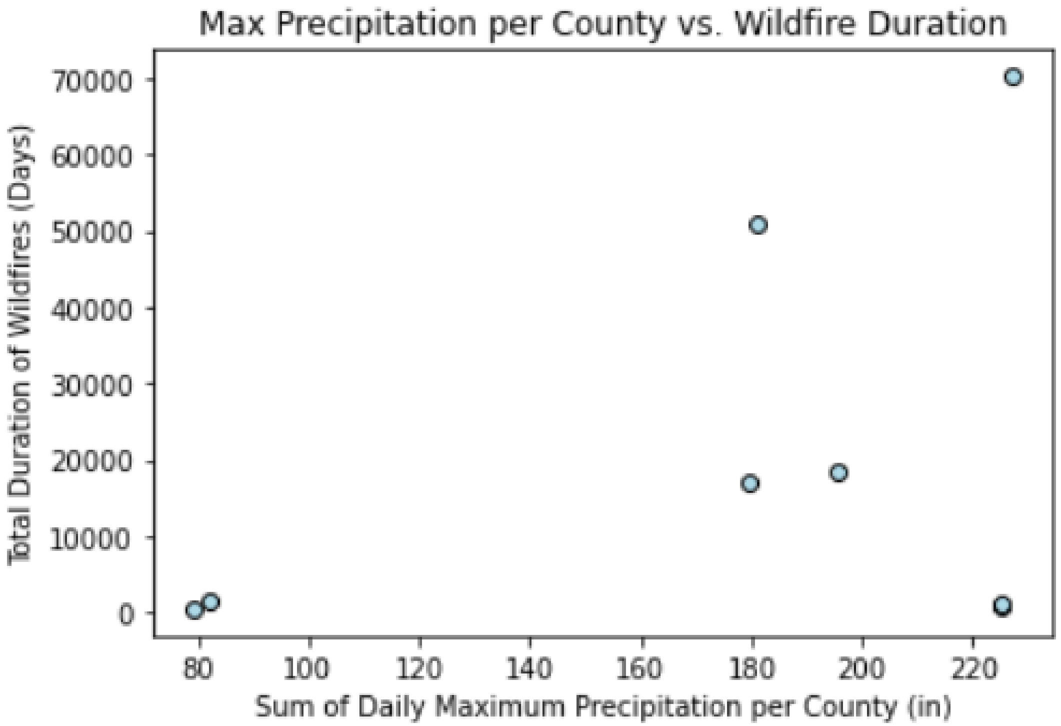
EMXP vs. Fire Extent

r-value: -0.47, weak correlation



EMXP vs. Fire Duration

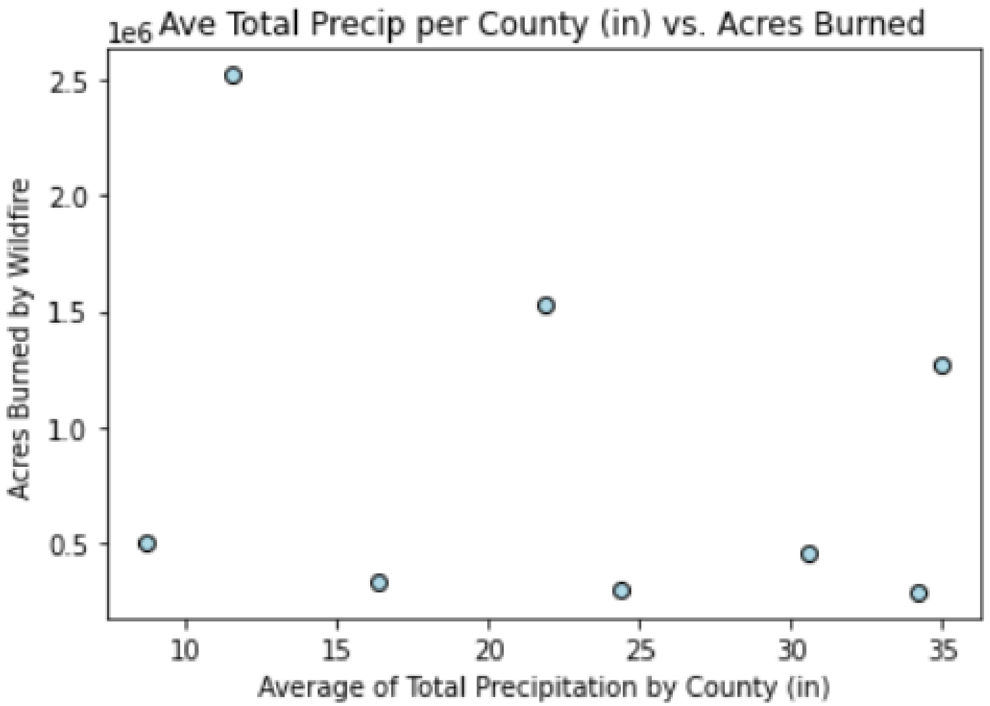
r-value: 0.39, weak correlation



RESULTS: EMXP

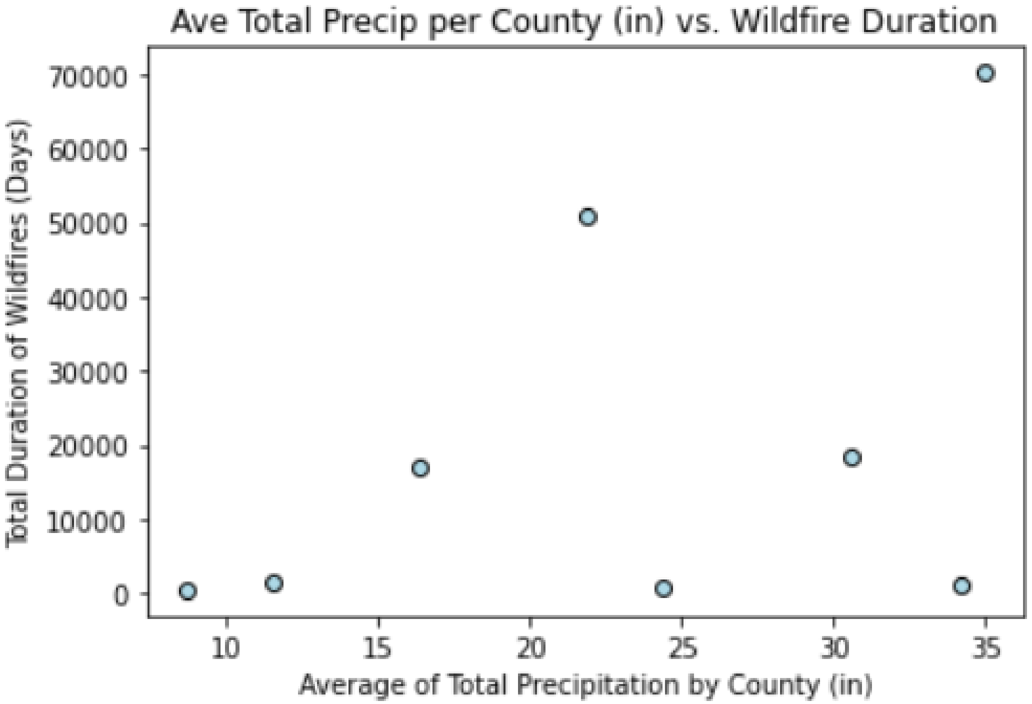
PRCP vs. Fire Extent

r-value: -0.29, very weak correlation



PRCP vs. Fire Duration

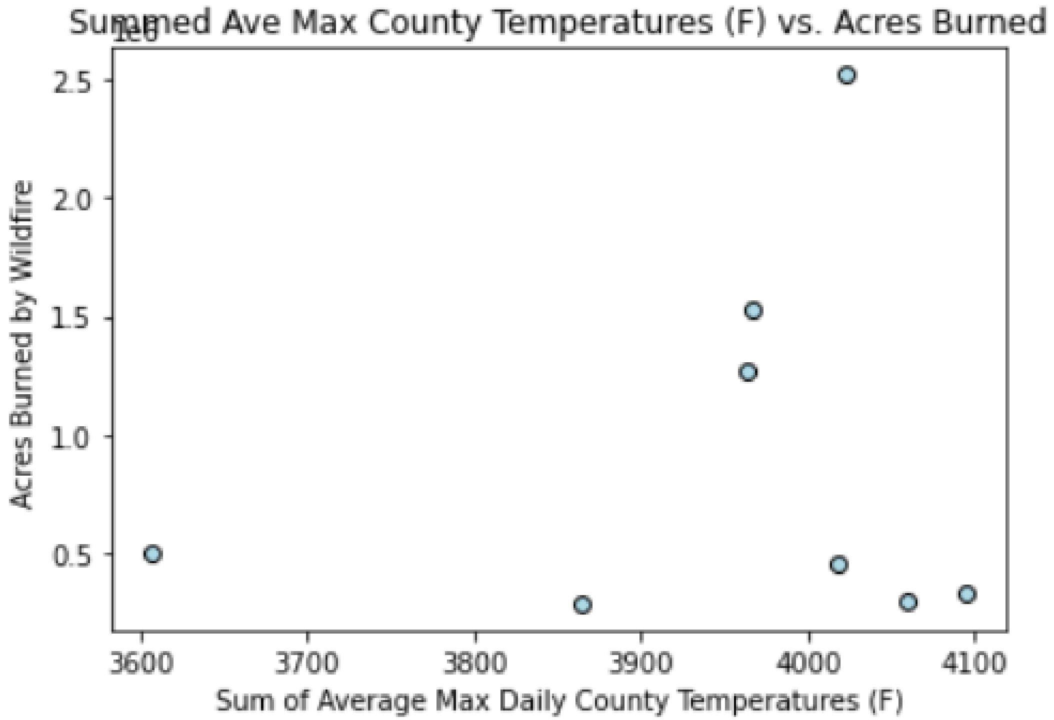
r-value: 0.44, weak correlation



RESULTS: PRCP

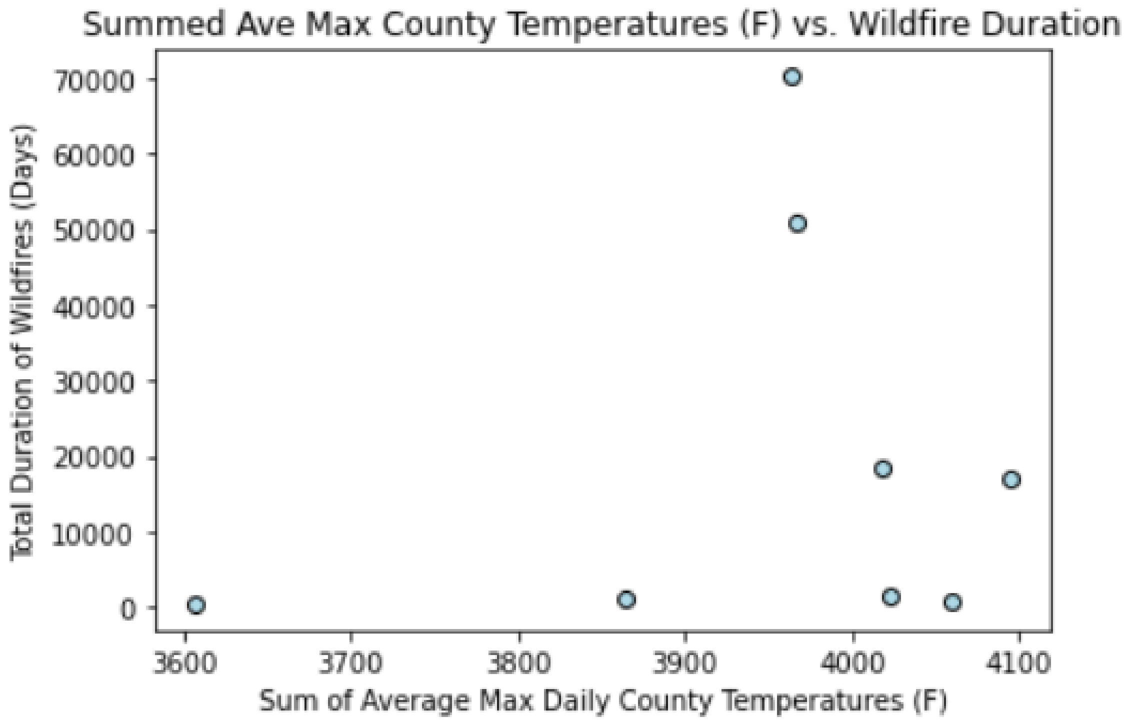
TMAX vs. Fire Extent

r-value: 0.17, no correlation



TMAX vs. Fire Duration

r-value: 0.19, no correlation



RESULTS: TMAX

DISCUSSION

- The results were not what I expected – I anticipated stronger correlations
- Potential Improvements:
  - Improved granularity of data – wildfire damage and duration at the county or city level versus at the state level
  - Visual data – heat map showing relationship between weather and fire
  - Additional statistical tests and manipulation of data
  - Additional years of data – ideally back to 1990
  - Analyze one year's fire data with the previous year's weather data – there could be a lag in weather impacts

QUESTIONS