CMSC 398z
Effective use of AI Coding Assistants and Agents

Bill Pugh and Derek Willis

Sept 26th, 2025

AI News story of the week

The great AI build out, and are we in a bubble?

Nvidia will invest $100B in OpenAI, which will then use that money to buy Nvidia products

On a recent conference call, Nvidia CEO Jensen Huang said that in this year alone, companies will spend $600B on AI data centers

- The entire US interstate highway system was built over 36 years for a total inflation adjusted cost of $300B.

Where is the revenue to pay for this going to come from?

csv files

csv - comma separated values

- need to handle issues such as a `,` `"` or *newline* in a cell value

Two standard ways to handle it in python

- Just return an array of strings for each row, including header
- Treat first row as headers, each following row is returned as a dict from column names to string values
  - Sometimes, a csv file can contain more than one table

All cell values treated as string, you need to parse them to get ints, floats, dates

Dataframe

A table where each column has a declared type (e.g., int, float, date, string)

In python, provided by pandas library

Often read/written from/to csv files

These types can be inferred, but you might want to specify them

- Handle cases such as int or none
- Error checking

Dates are always a problem in csv files. Excel is notorious for over eagerly interpreting numbers as dates

Dataframe operations

Dataframes are essentially an in memory database table

You can do things like filter or sort by a column, compute sums, etc

json - Javascript Object Notation

Every value in a json file is:

- a quoted string
- a number
- true or false
- null
- an Object: unordered list of key-value pairs, where the keys are strings
    - Enclosed in curly braces {}
- an Array: an ordered list of values
    - Enclosed in square brackets []

Understanding json files

Json files are typically distributed with no line feeds or indentation, which makes them very hard to understand

You can use a Json editor that can format json files, or provide a structured view

You can create a json schema, which describes the schema

json schemas are written in json, and the schema for json schemas is a json schema

Pydantic - Annotated Python classes with json capabilities

```
from pydantic import BaseModel


class User(BaseModel):

    name: str

    age: int

    email: str | None = None  # Optional field with a default value of None
```

Many programming languages have something similar, there are other ways to do this in python

XML - an older format, similar purpose as json

XML looks a lot like HTML - lots of angle brackets

Much stricter than HTML

Used for a lot of B2B APIs

You might encounter it, but might be able to get away with not dealing with it

Today's coding project

Analyze foreclosure data for Prince George's county

- public data, from 09/10/2009 until 09/15/2025
- includes date and address as well as other fields
  - Some of the address fields are mangled (some just random strings)
- 75,899 entries

```
"Tax Account Number","Property ID","Submitted Date","Street Address","Zip Code","City","State","Address Occupied","Property
     Description","Location"
"17060464123","129","09/10/2009","2625  Colebrooke, # 30 DR","20748","Temple Hills","MD","Unavailable","Unavailable","2625
     Colebrooke, # 30 DR Temple Hills MD,20748"
"2223246","130","09/14/2009","9001 3RD  AVE","20706","LANHAM","MD","Unavailable","Unavailable","9001 3RD  AVE LANHAM
     MD,20706"
"1839554","131","09/14/2009","2211 BANNING PL","20783","HYATTSVILLE","MD","Unavailable","Unavailable","2211 BANNING PL
     HYATTSVILLE MD,20783"
```

How might we analyze this data?

How might we analyze this data?

- Total # of foreclosures
  - by year or by month
  - trailing 12 month sum by month
  - by zip code
  - by year and zip code
- % of foreclosures
  - same breakdowns as above
  - Need # of housing units
- Get and analyze locations of foreclosures
  - geocoding all of those addresses will take a while, would likely need $
  - If you find a way to get precise lat/long for all of these addresses cheaply and quickly, let us know
- What other data would we need for this analysis?

Other sources of data

Number of housing units per zip code

- provided: Census bureau DP04, only data for 2020 provided - OK to just use this
- other information for zip codes, such as average income, etc also available

Location of each street

- provided: ArcGIS Online data for Maryland for Prince George county maintained roads
- doesn't provide information needed to map specific street addresses
  - but does give the latitude and longitude for points along each road
  - you can use this to get an approximate location for each address

How could we visual this data?

Text output

csv output

Graphs displayed as images

- There exist libraries that will help with this

Data overlaid on maps, viewable in web browser

- There exist libraries that will help with this

Cleaning up data

Missing zip codes: 1858

extracting street #, street name and kind from street address

- needed for looking up zip code
    - APIs available that will do this
- needed for looking up location of street

Data files you are provided with

County_Foreclosures.csv	Original list of foreclosures
County_Foreclosures_augmented.csv	Zip codes added to entries missing them
ACSDP5Y2020.DP04-Data.csv	Census DP04 data on PG 2020 housing units
Prince_George…..geojson	Data on locations of PG maintained roads
pg_county_boundary.geojson	Boundaries of PG county

Python code you are provided with

parse_address.py	Extract street #, name, kind and unit
geocoding_utils.py	Use OSM Nominatim API to get zip code
check_pg_county.py	Check if a lat,long is in PG county
augment_foreclosures.py	Add missing zip codes to County_Foreclosures.csv
test_*.py	Tests for the above, not all unit tests

Not all roads in PG are county maintained roads

The list of county maintained roads is not a complete list of roads in PG county

There are some regions with foreclosures but no county maintained roads

You can work to get additional information on road locations

The additional information I got using AI included lots of roads in the region that were not in PG county

which is why I wrote the code to determine if a point is in PG county

If you generate a map of each foreclosure location

My first attempt at this generated a file that was many megabytes in size

Slow to load and view in browser

Please don't submit many megabytes to submit server

  I believe it will be rejected

After I got a map I liked, I worked with the AI model to reduce the size

My own notes from working on this project

Parsing street addresses was challenging

- One of my mistakes: Didn't start by defining a bunch of unit tests
- In trying to handle a corner case, it wound up screwing up a lot more cases
  - lots of corner cases
- Didn't catch regression until later

Went down rabbit holes trying to get mapping locations for all PG roads

- and just roads in PG county

More notes

The ability to generate interactive maps and nice graphs of data was great

And I didn't have to learn anything about the libraries I was using

    Or about the javascript in the generated HTML

More comments

As I iterated on some methods, it kept on creating new methods

- Had to do lots of cleanup of duplicated or obsolete methods

If something isn't working, it loves to pursue ways to fix it, often taking a bad path

Didn't use git frequently enough to provide ways to backtrack

- The model's ability to backtrack was haphazard

At one point I was frustrated with the default copilot model, switched to Claude Sonnet 4

- Started getting throttled
- Tried to switch to using my Anthropic API Key, that wasn't working
  - exceeding input context

More comments

I spent several hours on this project

    yak shaving

The chat transcript was getting long, and the model was getting stupid as a result

I hadn't had a model maintain a TODO list, including completed items, so starting a new chat would have lost some important context