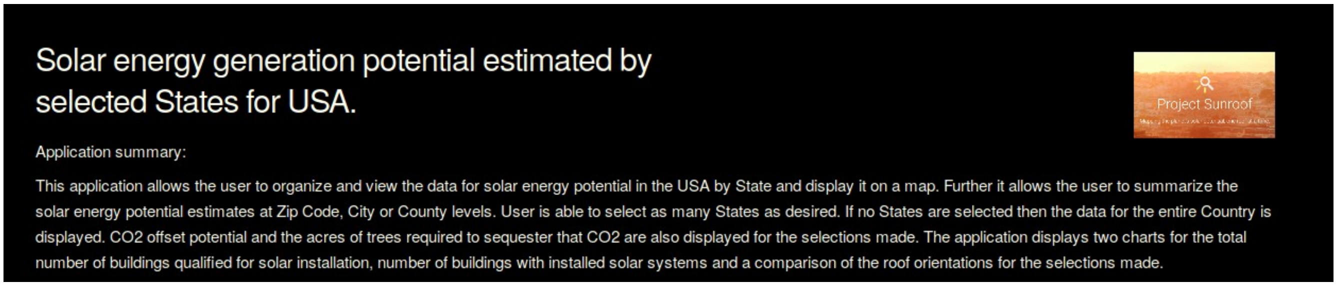
DASH

Link to the application: http://192.81.211.233:8050/

Introduction to App:

The app’s main objective is to show users the potential solar energy generation in their choice of states. The app also shows the carbon offset metrics in tons, number od building suitable for panel installations and already installed count of buildings. The data for this application is downloaded from https://www.kaggle.com/jboysen/google-project-sunroof Kaggle . The data is collected and made available by Google’s Sunroof project. Google uses their satellite map images to estimate the amount of the sunlight each building’s roof in US receives. They also have estimates for the potential solar energy generation for the roof facing in all 4 directions and flat roof.

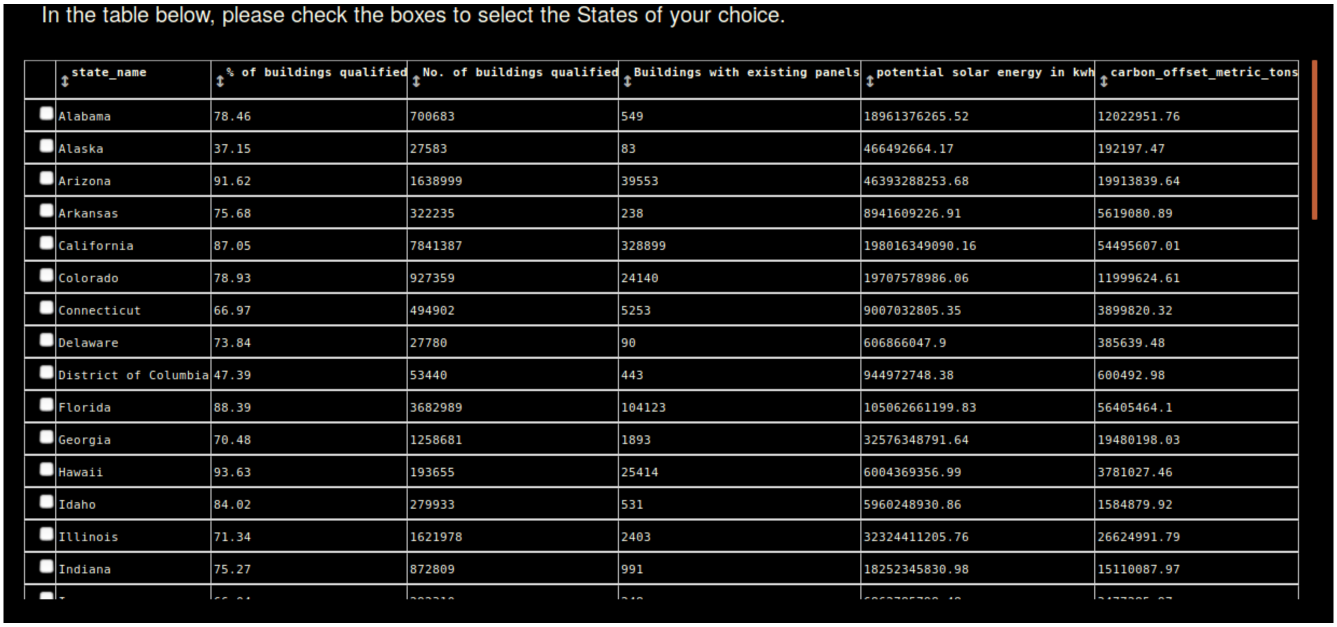


Solar energy generation potential estimated by selected States for USA.

Application summary:

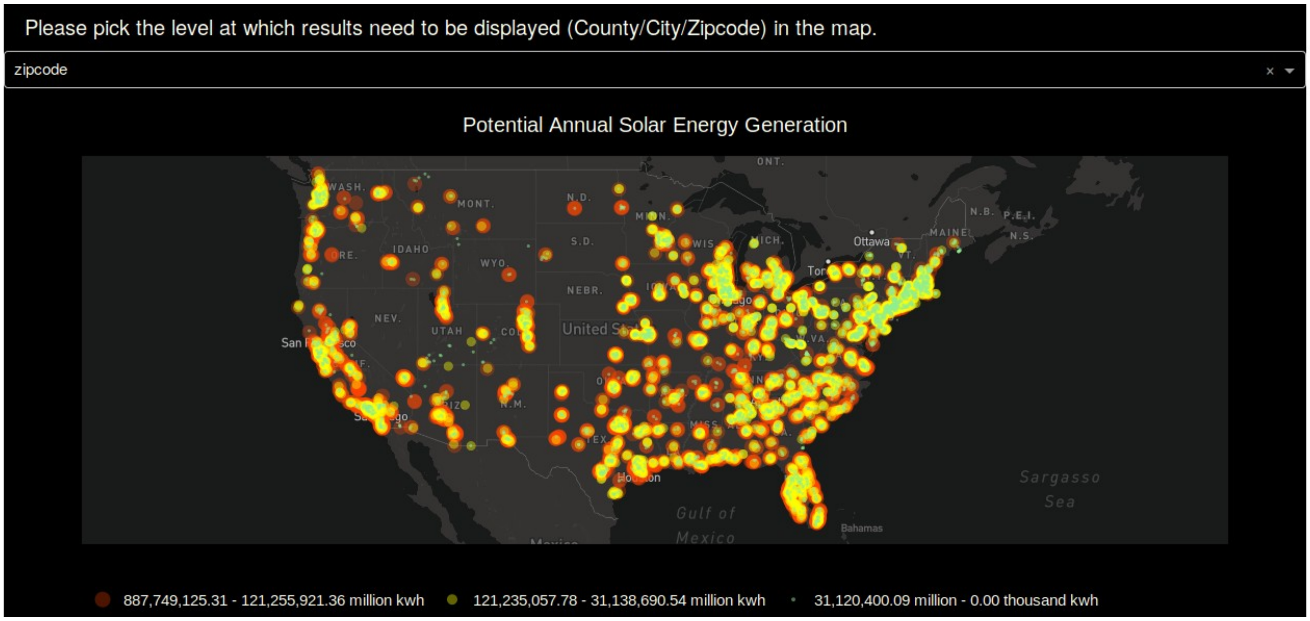
This application allows the user to organize and view the data for solar energy potential in the USA by State and display it on a map. Further it allows the user to summarize the solar energy potential estimates at Zip Code, City or County levels. User is able to select as many States as desired. If no States are selected then the data for the entire Country is displayed. CO2 offset potential and the acres of trees required to sequester that CO2 are also displayed for the selections made. The application displays two charts for the total number of buildings qualified for solar installation, number of buildings with installed solar systems and a comparison of the roof orientations for the selections made.

The layout of the application starts with a brief introduction of the app for the users. The app has 5 components in it, the first is the table which has all the states of the united states. For every state, table has percentage and number of the buildings qualified for solar panel installations, count of buildings with existing panels, estimate of potential solar energy generation and carbon-offset metrics in tons. The user is allowed to choose states of their choice, if no state is selected , all the plots will be showing the results for entire country. When multiple states are selected, the plots will be updated accordingly plotting for only those states. Same is for single state selection.



In the table below, please check the boxes to select the States of your choice.

	state_name	% of buildings qualified	No. of buildings qualified	Buildings with existing panels	potential solar energy in kwh	carbon_offset_metric_tons
□	Alabama	78.46	700683	549	18961376265.52	12022951.76
□	Alaska	37.15	27583	83	466492664.17	192197.47
□	Arizona	91.62	1638999	39553	46393288253.68	19913839.64
□	Arkansas	75.68	322235	238	8941609226.91	5619080.89
□	California	87.05	7841387	328899	198016349090.16	54495607.01
□	Colorado	78.93	927359	24140	19707578986.06	11999624.61
□	Connecticut	66.97	494902	5253	9007032805.35	3899820.32
□	Delaware	73.84	27780	90	606866047.9	385639.48
□	District of Columbia	47.39	53440	443	944972748.38	600492.98
□	Florida	88.39	3682989	104123	105062661199.83	56405464.1
□	Georgia	70.48	1258681	1893	32576348791.64	19480198.03
□	Hawaii	93.63	193655	25414	6004369356.99	3781027.46
□	Idaho	84.02	279933	531	5960248930.86	1584879.92
□	Illinois	71.34	1621978	2403	32324411205.76	26624991.79
□	Indiana	75.27	872809	991	18252345830.98	15110087.97



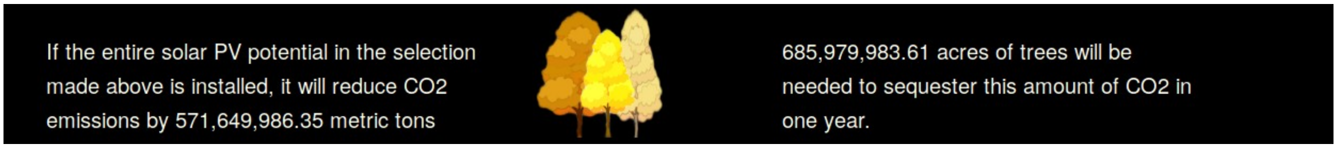
Please pick the level at which results need to be displayed (County/City/Zipcode) in the map.

zipcode

Potential Annual Solar Energy Generation

● 887,749,125.31 - 121,255,921.36 million kwh ● 121,235,057.78 - 31,138,690.54 million kwh • 31,120,400.09 million - 0.00 thousand kwh

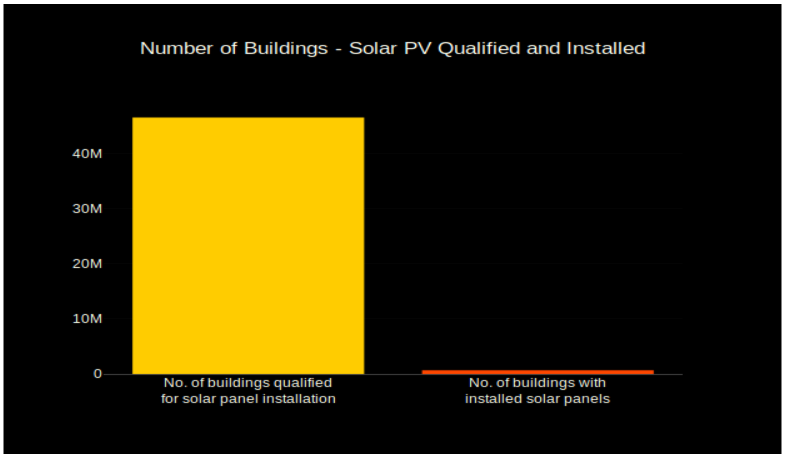
The second component is the scatter points on the map of the USA. The user has the option to choose from the county/city or zip-code level. For instance, if a user selects a particular state from the table, the user can then select city option from drop down menu. The map will show the location of all the cities in the selected states and when user hover the mouse over these points, they can read the potential solar energy generation value.



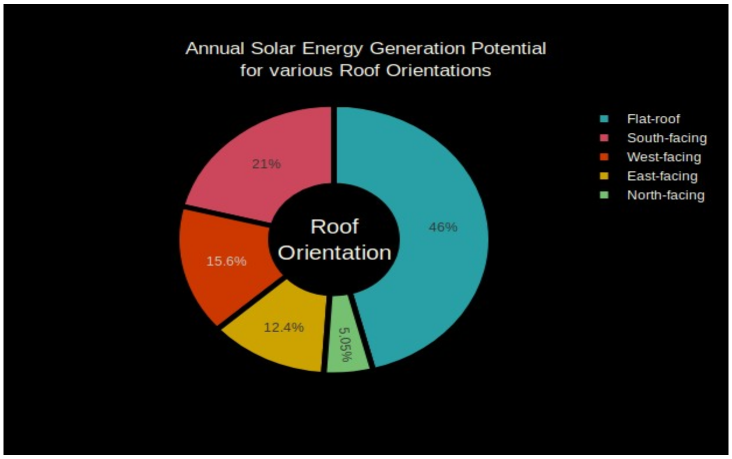
If the entire solar PV potential in the selection made above is installed, it will reduce CO2 emissions by 571,649,986.35 metric tons

685,979,983.61 acres of trees will be needed to sequester this amount of CO2 in one year.

The third component is a paragraph which gives the user quantity of carbon dioxide which can be reduced by installing solar panels in their selected region on all the buildings that are qualified for panels installations. To connect the user to the impact of CO2 production, I have included the number of acres of trees required to sequester the relevant quantity of CO2 in time period of one year. I found this equivalence using https://www.epa.gov/energy/greenhouse-gas-equivalencies-calculator website. As per this website for 1 metric tons of CO2 we will require 1.2 acres of trees to consume it. So I multiplied the value of CO2 produced with 1.2 for any selected states to produce the acres of trees required. The values of CO2 in metric tons and acres of trees are updated with the selection of the states.



Fourth component is the bar graph. This plots gives the users a comparison between number of building with and without solar panel installed in the selected state.



The last component of the app is the pie chart, this plots gives the user the percentage of potential solar energy generation on various roof orientations. The majority is always flat roof, for any selected states.

Developed by Bhavya.Ramgiri

This app is based on Google's Sunroof Project data from Kaggle.

This is the last part of my application, where I have provided my name and link to the source of the data provider.

Evaluation of the visualization in application using Kosslyn's principles:

1. As per the Kosslyn's Principles -**Relevance and Appropriate Knowledge**, I have explained briefly about what the app is and how the app works and have included link at the end of the app for user to visit the google sunroof project website for reference.
As per the the **Salience,Information Changes** and **Discriminability** principle, I have divided the scatter plot points in the map in three parts for e.g in county.csv file, it has 909 county information. I sorted them in descending order divide them into 3 equal parts gave each part a particular color and the divided scatter points also vary by size. These changes are displayed as legends on the map. I have also colored individual bars in the bar graph, this will draw the attention of the viewer and therefore the points and bars will be easily distinguishable.
These principle are the reason for me choosing black as my app's layout color instead of classic white. My intention was to bring out bright colors associated with sun light and with black background these bright colors stands out well and will have greater impact on viewers visual experience.
As per the **Perceptual Organization**, the pie charts has grouped together five values associated to single topic of roof orientation. This will be easy for viewers to group together these values with single idea.

There are two things in the app, which breaks rules laid out by Tufte and Kosslyn. As per the principles, it is better to avoid unnecessary items in the presentation which do not contribute to visualization of data. One is the google sunroof project logo and second is the trees image I have included in the carbon-offset paragraph. I believe that adding logo gives the viewers an impression that I am not entitled in deriving any conclusions using the app which I created. Further, trees image will create a visual imagination in viewers mind of the amount of trees required to sequester the CO2.

Development Process of the application:

My first idea at the time I started playing around dash tutorial codes was to mimic sunroof project website. I wanted users to input the state, county, city or zip-code and accordingly display as per the input the respective state map and the points on the particular city, county or zip-code mentioned by user. I was intimidated by this approach since I would have to call for each state map and loop all 51 states in callback. My understanding is the heavier the contents of the app is the slower it will be. If I had more time I would like to make my app as stated above. Next for all 4 data-set I wanted to create tabs each for state, city, county and zip-code, since all the 4 data-sets reflects the same idea, it would be redundant. So I settled in making the app which gives inclusive results.

Problems unsolved:

In the table of my app, I was unable to include the numbers separated with commas. It is indeed difficult for the users to read it, but I couldn't include commas as I was pulling the values from the table as per the selected states to plot my maps and graphs. However, I managed to reduce the decimal points to 2 digit number. Next I tried both pagination mode and scrolling vertically of my table. I found scrolling much easier than keeping on clicking next button in pagination mode to view all the states, even though scrolling is convenient, the column titles goes up from the view when scrolled down. I tried to use “n_fixed_rows=1” for fixing the column headers but the contents of the headers were overflowing to adjacent columns and letters were all lying on one another. I tried to constraint the width of the cell using pixels with all the possible number used in Dash Discussion but nothing worked.

Another issue I faced is with the legends of the scatter map plot. My app allows user to select one out of three options to display points in map. The legends do not update as the scatter points do for each selection, I have to click on the legends and then it changes.

Additionally the tree image that I have included is not being placed at the middle of the page, in the layout of “app.py” I have specified it between the two paragraphs and have assigned it equal columns in page layout as per styling sheet.

Problems solved:

Recently I found that the original zipcode.csv file had zip-code in the decimal for e.g “12548.0” and the same was displaying in the scatter map’s hover text. I took help of bash’s sed command “sed 's/\.0//' zipcode2.csv > zipcode2.csv” to substitute “.0” in every line to nothing. The sed command only substitutes the first occurrence in each line so I don't have to worry about any other numbers in my csv data file having “.0” in it. I also found that the python was not considering the zeros of zipcode like “08310”. So I had to include “{:05}.format(zipcode.region_name)”, zipcode being pandas data frame reading data from zipcode2.csv and region_name is where zip code is saved.

When styling the table and some widget I encountered that the widget were not changing the color when I set their background color to black. Ultimately with your help I was able to track the component's class name using web developer inspector and change the color of every section of app. However, the small rectangular box where ticks are added in the table when inspected had no class name, so I was not able to change its color.