**PREPARED BY**

Priyanka Soni

Yash Choksi

|  |  |
| --- | --- |
| Deep Solar project  CIS-5250 : Visual Analytics | Abstract  Using the Deep learning Data to understand adaptability of Solar Panel installations. |

|  |  |
| --- | --- |
| A | Introduction |
| B | Data Cleaning |
| C | Show/Apply Summary Statistics |
| D | Analysis & Visualizations |
| E | Conclusion |
| F | Appendix |

# **Introduction**

**Data set URL’s and Data set Description**:

The dataset we have chosen is a raw product of deep learning project conducted by Stanford University engineers that used Satellite Imagery to identify location and size of the Photo Voltaic panels across the US. It contains of approximately 73K rows and 169 Columns consisting of information about, average household income, Number of tiles/solar system(solar panels) installed , size of installation etc. in residential as well as non-residential areas for all the counties/states.

Further information for the dataset can be found as below:

Dataset URL :

<https://www.kaggle.com/tunguz/deep-solar-dataset>

Information on Deep solar Project:

<http://web.stanford.edu/group/deepsolar/home>

We would be using this dataset to do the exploratory analysis and create visualization using python. For the purpose of this project, we have narrowed down the data set based on states which are as following:

CA – California

TX – Texas

NY – New York

FL – Florida

NV – Nevada.

The idea is to have dataset that represents a mix of data from various states that have different weather conditions and see the performance overall.

Within these, in order to do exploratory analysis, we would be using some of the columns mentioned below:

Household income

Population density

Solar system installation – Residential

Solar system installation – Non-Residential

Education

Frost days

County

Electricity Price

Solar System Count

1. **Data Cleaning**

As we have a huge amount of dataset available, it was very important for us filter out the data in order to create meaningful visuals and extract insights from these. We started with importing the packages Panda and Numpy and using the panda dataframe we read the raw csv file named *before\_cleaning* that was extracted from Kaggle.

A screenshot of a cell phone

Description automatically generated

Figure

As we wanted to narrow down dataset based on states, we first did a count from each state as to get the availability of rows in each state using the function value\_counts().

A screenshot of a cell phone

Description automatically generated

Figure

As we wanted to bring down the dataset size to 1500 number of rows to create visuals, we chose 5 states with different weather conditions and created a new dataframe to study. In order to do so, we fetched the data state wise and create a separate dataframe for each state and reset the index to default i.e 0.

A screenshot of a cell phone

Description automatically generated

Figure

As this is not our final dataset and we would be combining these later, we drop the index column and the updated dataframe is as below:

A screenshot of a cell phone

Description automatically generated

Figure

We examined that the data when fetched is in sorted order which might result in having skewed dateframe. Hence to avoid that, we shuffle the data and extract 300 rows from each using random function and for loop. A screenshot of a cell phone

Description automatically generated

Figure

As we can see from fig.1 California has 8055 rows. In the above code, for loop uses range function starting from 0 – 300 to extract shuffled data for California from earlier filtered data frame named *californiaFinal*.

We repeat this exercise for other states as well (i.e NY, TX, FL,NV) and create filtered & shuffled data frame having 300 rows in each state. Once this is done, we combine these using append() method and the new dataframe is generated called *finalDataSet*  having 1500 rows in total.

A screenshot of a cell phone

Description automatically generated

Figure

A screenshot of a cell phone

Description automatically generated

Figure

One of the columns that we would be using for visualization is frost\_days, which consists of few null values. Using fillna() function , we replace the null values with mean values of frost\_days to have consistency in the data.

A screenshot of a cell phone

Description automatically generated

Figure

1. **Summary statistics**

Using the .describe() method in pandas, we would like to see summary statistics for columns shown below:

A screenshot of a cell phone

Description automatically generated

One of the columns we have used here is gini\_index which basically used as a gauge of economic inequality, measuring income distribution or, less commonly, wealth distribution among a population. The coefficient ranges from 0 (or 0%) to 1 (or 100%), with 0 representing perfect equality and 1 representing perfect inequality. As we can see, the mean gini index for our data set is 0.421346 (42.13%) which indicates that income distribution is not bad in our filtered dataset for the 5 states.

Similarly we have summary statistics for various other columns such and some of them we would be using for Visualization and analytics to understand them in detail.

1. **Analysis & Visualizations**

Now that we have the clean dataset, we utilize them to get insights.

1. **Total solar panel area in Residential as well Non-Residential area**

In order to understand what is the acceptance rate of solar panels Residential vs Non-residential areas we use the columns *total\_panel\_area\_residential & total\_panel\_area\_nonresidential* .

Python code :

From the final dataset , we first calculate sum of the total panel area for both the categories using method sum()

A screenshot of a social media post

Description automatically generated

Figure

Once we have the sum values, we plot the graph using matplotlib package.

A screenshot of a cell phone

Description automatically generated

A screenshot of a social media post

Description automatically generated

Figure

A screenshot of a cell phone

Description automatically generated

Figure

As seen, the Total Solar panel area installation is higher in Non residential area which mainly consists of industries and other commercial firms. The main concern over solar power is the initial investment of purchasing and installing the panels. Even when government subsidies are available, the cost isn't feasible for many homeowners, thus the reason why solar power isn't more widely used today in residential areas.

1. **Effect of weather on Solar Panel installations.**

As we have mix bag of data from various states, next aim would be to understand what if there is any correlation between weather conditions and usage of solar panels overall. The columns frost\_days and solar\_system\_count i.e total count of installations for residential & non-residential.

Python code :

A screenshot of a cell phone

Description automatically generated

Figure

Since we want to understand the correlation between these two variables, scatterplot would be best way to explain that.

A close up of a white wall

Description automatically generated

Figure

It is pretty evident that as number of frost days increases, the rate of solar system installation decreases. Due to lower days of sun visibility people are not very motivated to install these panels as they feel it would not serve the purpose during snow days which would most likely be NY.

1. **Average Household Income**

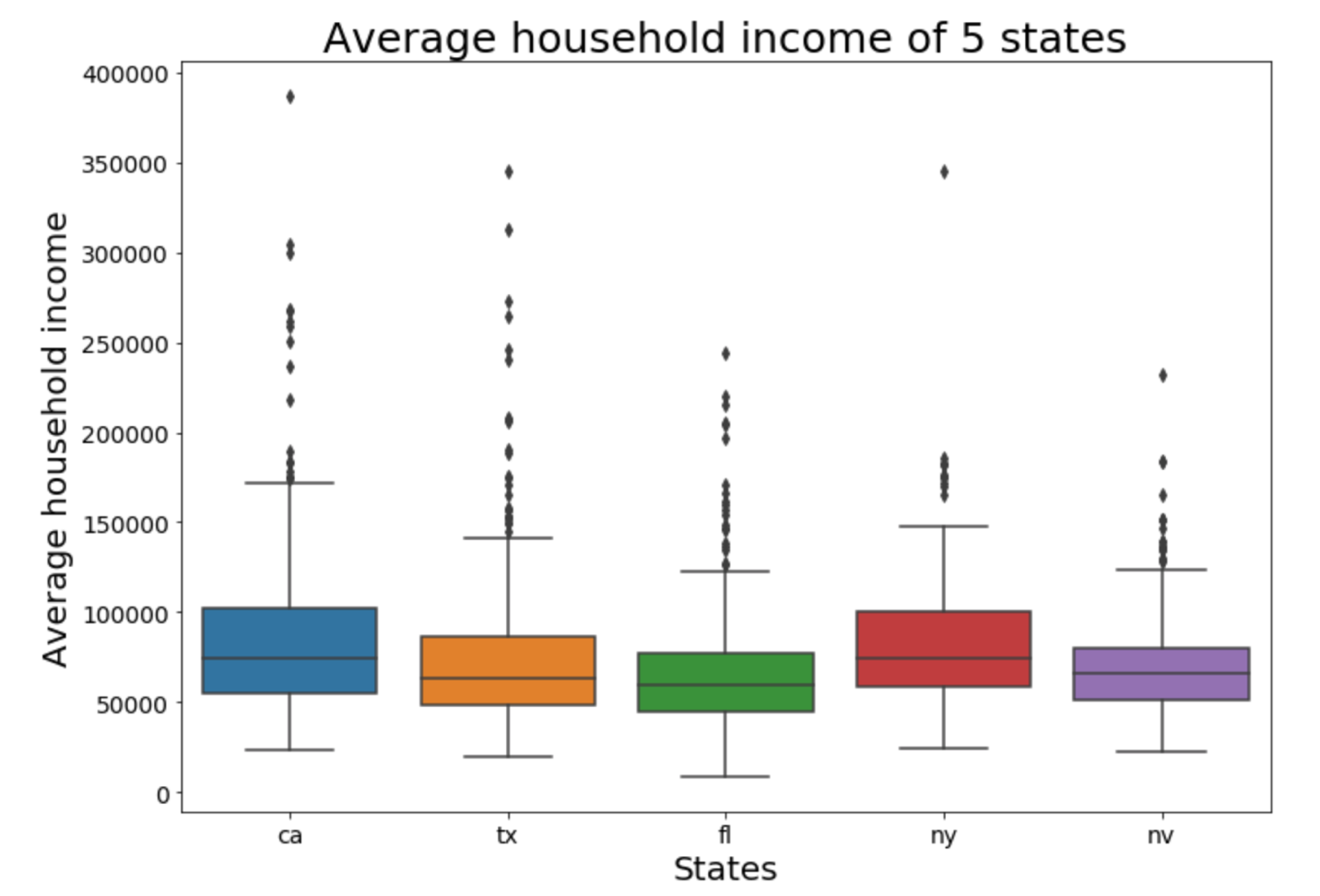
Apart from weather conditions, we feel income is also one of the important factors for installations of solar systems panels. To study this, we use the column avg\_household\_income in each state.

Python code:

A screenshot of a cell phone

Description automatically generated

Firstly, we observed that the household\_income column has some null values, so we replaced those by sum of avg values so we have a complete data. Since we wanted to show avg income across all the states and what are the threshold values among them, we used Boxplot as it would good indication of how the values in the data are spread out. As we are using multiple box plots, we use the method subplot() that would help us in getting all the graphs together.



The above results show that California has the highest avg. household income with a median value of approx. $75,000 annually and highest income of approx. $175,000 and a minimum of $25,000. California is followed by New York, Texas , Nevada and the lowest is for Florida.

1. **County with Highest number of solar panel installations.**

We further narrow it down to county and would like to know which county has the highest installation of solar panel.

Python code:

A screenshot of a cell phone

Description automatically generated

We first create empty dictionary of count of solar system installed county wise called *numberofSolarSystemsCounty* . The create another dictionary called *countyDict* that counts county values and adds that to empty dictionary created earlier using *.to\_dict()* method. Once that is done, we create a list called *countyList* which stores all the Keys from the *countyDict* dictionary.

A screenshot of a cell phone

Description automatically generated

Using for loop, we use variable data1 to extract the rows with number of solar system installations in all the counties and add that to our original dictionary *numberofSolarSystemsCounty* using the .update() method. Once we have the data extracted, we sort the data to get top 5 counties using *sorted* function and this data is then stored in *sortedCountyData* list.

To plot the graph for Top 5 counties with solar system installations , we use lineplot from seaborn package.

A screenshot of a cell phone

Description automatically generated

As seen in above figure, Clark county in Nevada has the highest number of installations which is obvious as it is the desert area and has maximum exposure to sun throughout the year. In fact there is huge solar farm which is a government initiative to focus more towards clean energy. After Nevada, the other 4 counties fall under California state which can also be linked to highest average household income compared to other states that is shown in previous Visual.

1. **Education**

Now that we have insights on effect of weather, income on number of installations, we would like to see if there is any relation of education on number of installations.

Python code:

A screenshot of a social media post

Description automatically generated

Once again , we make use of scatter plot as we would like to see if education has any correlation with installation of number of solar system installations and for that we would be using education\_population and solar\_system\_count columns.

A screenshot of a social media post

Description automatically generated

As we can see with increase in educated population, the installation of total solar panels increases. There is major chunk in the range 0 – 6000 of educated population, meaning out of the population density of each state on an avg only 2000-4000 people have completed their graduation and have fewer installations of solar panels. .This reflects that there might be a segment of population that might not be aware of the benefits of using clean energy or if there are incentives available on using these systems for daily usage. It could be a used as a separate project to understand if there is any gap in people’s perspective on benefits of solar system installations.

**Python Code:**

import pandas as pd

import numpy as np

# Main csv file

data = pd.read\_csv('./before\_cleaning.csv')

# Statewise values

data.state.value\_counts()

# We are takinng 5 states in consideration

# States are California, New York, Florida, Texas, Nevada

# Fetch specific data with state have value as Ca=Califorina

california = (data[data.state=='ca'])

# Reset index from 0.

california = california.reset\_index()

# Drop newly un-necessary created index column

california = california.drop(columns=['index'])

california.head()

# # Fetch specific data with state have value as ny=New York

newYork = (data[data.state=='ny'])

# # Reset index from 0.

newYork = newYork.reset\_index()

# # Drop newly un-necessary created index column

newYork = newYork.drop(columns=['index'])

newYork.head()

# Fetch specific data with state have value as Ca=Califorina

texas = (data[data.state=='tx'])

# Reset index from 0.

texas = texas.reset\_index()

# Drop newly un-necessary created index column

texas = texas.drop(columns=['index'])

texas.head()

# Fetch specific data with state have value as Ca=Califorina

florida = (data[data.state=='fl'])

# Reset index from 0.

florida = florida.reset\_index()

# Drop newly un-necessary created index column

florida = florida.drop(columns=['index'])

florida.head()

# Fetch specific data with state have value as Ca=Califorina

nevada = (data[data.state=='nv'])

# Reset index from 0.

nevada = nevada.reset\_index()

# Drop newly un-necessary created index column

nevada = nevada.drop(columns=['index'])

nevada.head()

# Here are list of columns and data is manipulated in form of list.

columns = list(data.columns)

# Create empty dataFrame for each 5 states for new data

californiaFinal = pd.DataFrame(columns=columns)

texasFinal = pd.DataFrame(columns=columns)

newYorkFinal = pd.DataFrame(columns=columns)

floridaFinal = pd.DataFrame(columns=columns)

nevadaFinal = pd.DataFrame(columns=columns)

# Califorrnia shuffled dataframe using random number to not get skewed data.

# Generate random number module

import random

# random.randint(0, 8055)

for x in range(0, 300):

number = random.randint(0, 8055)

californiaFinal = californiaFinal.append(california.iloc[number], ignore\_index=True)

californiaFinal.head()

# New York shuffled dataframe using random number to not get skewed data.

import random

# random.randint(0, 8055)

for x in range(0, 300):

number = random.randint(0, 4917)

newYorkFinal = newYorkFinal.append(newYork.iloc[number], ignore\_index=True)

newYorkFinal.head()

# Texas shuffled dataframe using random number to not get skewed data.

import random

# random.randint(0, 8055)

for x in range(0, 300):

# Generate random number everytime in for loop

# Generate random number everytime due to geting not skewed data

number = random.randint(0, 5265)

# Getting random rows for not getting random rows

texasFinal = texasFinal.append(texas.iloc[number], ignore\_index=True)

texasFinal.head()

# Florida shuffled dataframe using random number to not get skewed data.

import random

# random.randint(0, 8055)

for x in range(0, 300):

# Generate random number everytime in for loop

# Generate random number everytime due to geting not skewed data

number = random.randint(0, 4244)

# Getting random rows for not getting random rows

floridaFinal = floridaFinal.append(florida.iloc[number], ignore\_index=True)

floridaFinal.head()

# Nevada shuffled dataframe using random number to not get skewed data.

import random

# random.randint(0, 8055)

for x in range(0, 300):

# Generate random number everytime in for loop

# Generate random number everytime due to geting not skewed data

number = random.randint(0, 686)

# Getting random rows for not getting random rows

nevadaFinal = nevadaFinal.append(nevada.iloc[number], ignore\_index=True)

nevadaFinal.head()

# Merging all dataframe in finalDataSet

finalDataSet = californiaFinal.append(texasFinal).append(floridaFinal).append(newYorkFinal).append(nevadaFinal)

finalDataSet = finalDataSet.reset\_index()

finalDataSet = finalDataSet.drop(columns=['index'])

finalDataSet.head()

# Equal distribution is created.

finalDataSet.state.value\_counts()

gini = finalDataSet.gini\_index

electricityPrice = finalDataSet.electricity\_price\_overall

import matplotlib.pyplot as plt

import seaborn as sns

# fig, ax = plt.subplots()

sns.jointplot(gini,electricityPrice, kind='hex')

finalDataSet.total\_panel\_area\_residential.sum()

finalDataSet.total\_panel\_area\_nonresidential.sum()

panelAreasValues = [finalDataSet.total\_panel\_area\_residential.sum(), finalDataSet.total\_panel\_area\_nonresidential.sum()]

panelAreasKeys = ['Total panel area residential', 'Total panel area non-residential']

fig3, ax3 = plt.subplots()

ax3 = sns.barplot(panelAreasKeys, panelAreasValues, palette='Blues\_d')

ax3.set\_xlabel('Category', fontsize = 25)

ax3.set\_ylabel('Total solar panel area', fontsize=25)

ax3.tick\_params(labelsize=18)

fig3.set\_size\_inches(11.7, 8.27)

ax3.set\_title('Solar panel areas in residential and non-residential', fontsize=25)

plt.show()

frostDaysMean = int(finalDataSet.frost\_days.mean())

finalDataSet.frost\_days = (finalDataSet.frost\_days.fillna(frostDaysMean))

fig2, ax2 = plt.subplots()

ax2 = sns.scatterplot(finalDataSet.frost\_days, finalDataSet.solar\_system\_count, marker='+')

ax2.set\_xlabel('Frost days', fontsize=20)

ax2.set\_ylabel('Total solar systems', fontsize=20)

fig2.set\_size\_inches(11.7, 8.27)

ax2.tick\_params(labelsize=18)

ax2.set\_title('Frost days to total solar system', fontsize=25)

plt.show()

finalDataSet.daily\_solar\_radiation = finalDataSet.daily\_solar\_radiation.fillna(finalDataSet.daily\_solar\_radiation.mean())

# Top 5 county with maximum number of solar systems

numberOfSolarSytemsCounty = dict()

countyDict = (finalDataSet.county.value\_counts()).to\_dict()

countyList = list(countyDict.keys())

for data1 in countyList:

counter = 0

for idx, row in finalDataSet.iterrows():

if row.county == data1:

counter = counter + row.solar\_system\_count

numberOfSolarSytemsCounty.update({data1:counter})

# lambda is function implementation.

sortedCountyData = sorted(numberOfSolarSytemsCounty.items(), key=lambda x: x[1], reverse=True)

# Geetting first 5 data with maximum number of solar systems

finalCountyData = sortedCountyData[0:5]

countyName = []

countyValues = []

for x in range(0, 5):

countyName.append(finalCountyData[x][0])

countyValues.append(finalCountyData[x][1])

fig1, ax1 = plt.subplots()

ax1 = sns.lineplot(countyName, countyValues)

ax1.set\_xlabel('County names', fontsize = 20)

ax1.set\_ylabel('Number of solar systems', fontsize=20)

fig1.set\_size\_inches(11.7, 8.27)

ax1.tick\_params(labelsize=14)

ax1.set\_title('Top 5 county with highest number of solar systems', fontsize=25)

plt.show()

# ax.set(ylim=(1344000, 1635000))

finalDataSet.average\_household\_income.isna().sum()

finalDataSet.average\_household\_income = finalDataSet.average\_household\_income.fillna(finalDataSet.average\_household\_income.mean())

averageHouseholdIncome = finalDataSet.average\_household\_income

stateNames = ['California', 'Texas', 'Florida', 'New York', 'Nevada']

fig5, ax5 = plt.subplots()

ax5 = sns.boxplot(x=finalDataSet['state'], y=finalDataSet['average\_household\_income'])

ax5.set\_xlabel('States', fontsize = 20)

ax5.set\_ylabel('Average household income', fontsize = 20)

fig5.set\_size\_inches(11.7, 8.27)

ax5.tick\_params(labelsize=14)

ax5.set\_title('Average household income of 5 states', fontsize=25)

plt.show()

finalDataSet.gini\_index = finalDataSet.gini\_index.fillna(finalDataSet.gini\_index.mean())

giniIndex = finalDataSet.gini\_index

fig6, ax6 = plt.subplots()

ax6 = sns.scatterplot(finalDataSet.education\_population, finalDataSet.solar\_system\_count)

ax6.set\_xlabel('Educated population', fontsize=20)

ax6.set\_ylabel('Total solar systems', fontsize=20)

fig6.set\_size\_inches(11.7, 8.27)

ax6.tick\_params(labelsize=18)

ax6.set\_title('Educated population to number total solar systems', fontsize=25)

plt.show()

# ax2.set\_title('Frost days to total solar system', fontsize=25)

summaryStats = finalDataSet[['state', 'solar\_system\_count', 'education\_population', 'county', 'gini\_index', 'electricity\_price\_overall', 'solar\_system\_count', 'frost\_days', 'total\_panel\_area\_residential', 'total\_panel\_area\_nonresidential', 'average\_household\_income']]

summaryStats.describe(include='all')

summaryStats1 = finalDataSet[['state', 'solar\_system\_count', 'education\_population', 'county']]

summaryStats1.describe()