Using Census Data to Predict Solar Energy System Density per Census Tract

Eddie Sun: edsun@stanford.edu
Jeremy Chen: jchen@stanford.edu
Brett Szalapski: brettski@stanford.edu

November 19, 2018

1 Motivation

New renewable energy sources require improvements to the current electric grid. The recent surge in the number of intermittent energy generation facilities, such as solar and wind farms, has resulted in a need for improved monitoring and control methods for the electric grid due to increased supply-side uncertainty. Mitigating this uncertainty in the amount of electricity produced would greatly increase power generation efficiency.

One major component of supply-side uncertainty comes from residential solar panel installations. Today, installing solar panels on residential homes is already easy and affordable, and will only get easier and cheaper as time progresses. As a result, it is difficult to know how many solar panels exist and supply power to the grid. If energy companies had more insight into this piece of the supply-side puzzle, they could better model an area's energy production and balance power plant production accordingly, resulting in lower energy costs and less environmental impact.

2 Dataset

The dataset has three unique labels —the number of solar panels in each census tract, the number of solar tiles, and the total tile area —along with around 180 demographic statistics of each respective census tract[00]. The labels were determined from a previous project originating from ES's research group, which used deep learning to count solar panels from satellite images[0]. The dataset contains around 75,000 data points. Using this dataset, we plan to create a supervised-learning algorithm using census data to predict the number of solar panels or solar systems in each census tract.

3 Previous Work

There are surprisingly few previous published studies that use existing census data to make demographic predictions compared to the amount of census data available; the few that were found are described here. The University of California, Irvine's (UCI) 1996 census income dataset has been used to predict whether income levels are below or above $50,000/yr using logistic regression and random forests, achieving classification accuracy of around 92%[1]. A previous CS229 project accomplished the same task with the same dataset, also using random forests and logistic regression[2].

Neural networks have also been used in conjunction with census data. Wang et al predicted population counts using a standard feed-forward neural network with 4-6 hidden layers[3], however the main focus of this work was to compare the neural network performance with that of linear regression. Census and weather data have also been used to augment crime data to forecasts crime in Chicago and Portland as a multi-classification problem[4]. The authors accomplished this with several neural network architectures: feed-forward, convolutional, recurrent, and recurrent convolutional, with the best result being about 75% accuracy. A third deep-learning study predicted age and gender from cell-phone metadata using a convolutional network[5].

To our knowledge, the soon-to-be-published paper from which the dataset is obtained is the first to utilize machine learning for large-scale surveying of solar panels. Near-term solar power forecasting using machine learning is more commonplace[6], but this project and the aforementioned paper are among the first to study system installation counts, which are more correlated with long-term solar power forecasting and market sizing.

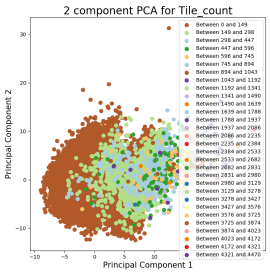
4 Methods

Since previous published work using census data to make predictions is somewhat scarce, we created two different machine-learning algorithms: support vector regression (SVR), and a fully connected neural network (NN). These two algorithms were chosen because both can learn highly-nonlinear trend-lines, and we do not suspect the data is linear. For the SVR algorithm, we performed principal component analysis (PCA) in order to decrease the number of input features. As of this milestone, both models have been run against subsets of our dataset with promising results. The following sections detail the PCA, SVR, and Neural Network progress.

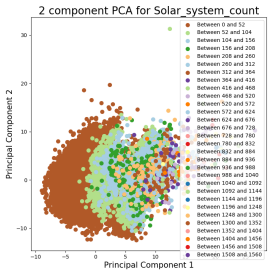
4.1 PCA

With 180 features to build a model on, we want to explore the possibility of reducing the number of features of the dataset used for training the models. The first implementation for doing so is Primary Component Analysis. The initial goal was to reduce the number of features to two primary components so that

the data could be plotted and examined for any obvious trends. Figures 1a, 1b, and 2 show that Tile Area may be the least productive label to use. The strongest trend is in the System Count label, so initial efforts will be focused on that label.



(a) Tile Count



(b) System Count

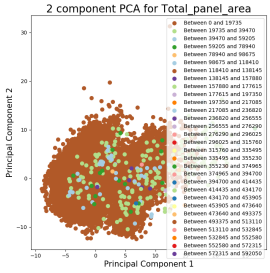


Figure 2: Tile Area

4.2 SVR

One of the first experiments carried out was an SVR model of the data. This simple model establishes a baseline for our future experiments, and given the promising results of the linear kernel implementation, could prove to be one of the strongest models.

We implemented a kernelized-SVR with three different kernels: the third-degree polynomial kernel, the linear kernel, and the RBF kernel. Each model is trained under 604 examples and the average R2 is validated against a 70-example validation set. In the SVR model, the penalty parameter is set at 1000, kernel cache size is 200, and the kernel coefficient is set at $\frac{1}{n\sigma_x}$ where n is the number of features. The results of these models are compared in Figure 3. Based on the prediction error on the validation set, SVR using the linear kernel has much better accuracy compared to the RBF polynomial kernels. Model selection and parameter tuning was performed with SciKitLearn's GridSearchCV module.

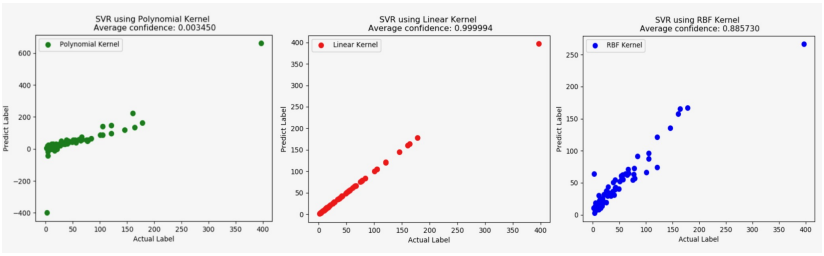


Figure 3: Comparison of SVR Kernel Performance

4.3 Neural Network

For the milestone, the goal was to simply have the neural network run without errors and make a prediction with reasonable accuracy. This was accomplished.

The neural network is a standard feed-forward NN coded using `Keras`. Currently, the model consists of 2 hidden layers with 64 neurons each, ReLU activation functions for each layer, RMSprop as the optimizer, a learning rate of 0.001, batch size of 64, and is trained for 100 epochs. For debugging purposes, the model was only trained on 750 examples split 80/10/10 for train, dev, and test. Results are shown in Figure 4. Note that optimization will be performed between the Milestone and the final report, but the network shows promising initial results.

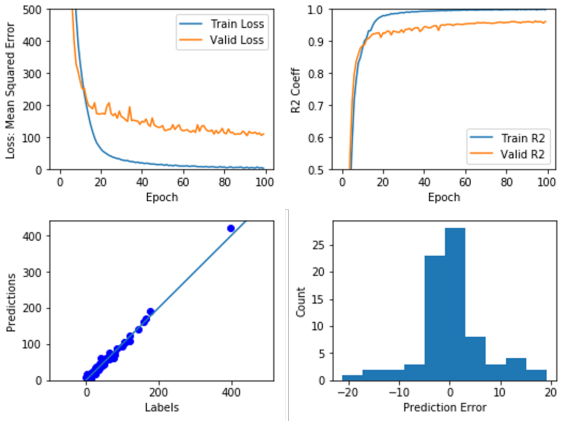


Figure 4: Neural Network Performance: **Top left:** L2 loss on the train and dev data. **Top right:** R^2 coefficient of the trend line on the train and dev data. **Bottom left:** Predictions made by the network on the test data features plotted against the test data labels. **Bottom right:** Histogram of the error (difference) between the model predictions and the true test labels.

5 Future Work

Each of the three areas has ongoing work. More conclusions regarding which features to use and which to eliminate will come from PCA analysis, which has yet to be used to pare down which features are fed into the other models. Furthermore, there are a couple of categorical columns, such as state, which have not be used yet. We may look to add these columns.

The SVR implementation will need to be run against the entire dataset and retuned with the larger train and dev sets accordingly. Kernel selection will be reevaluated as well.

The Neural Network hyperparameters will need tuning along with using the rest of the dataset. The network will assist with feature selection as well using numerical gradients.

Finally, our team is seeking a direction to take our project beyond a basic application project, *something on which we would appreciate feedback and guidance.*

Contributions

- Eddie: Neural network implementation, write-up, research on previous work
- Jeremy: SVR implementation and write-up
- Brett: PCA implementation and write-up, data parsing and preprocessing, document formatting and editing

References

[00] American Community Survey. 2015. 2015 ACS 1-year estimates [data file]. Retrieved from http://factfinder.census.gov

[0] Yu, Jiafan, et al. "DeepSolar: A Machine Learning Framework to Efficiently Construct Solar Deployment Database in the United States." Joule (2019), accepted.

[1] Sheremata, Jeff. "Machine Learning Income Prediction Using Census Data." Medium.com, Medium, 11 Jan. 2017.

[2] Voisin, M. Prediction of Earnings Based on Demographic and Employment Data.CS 229 Report, 2016.

[3] Wang, Jun, et al. "Advances in Neural Networks - ISNN 2006." Lecture Notes in Computer Science, May 2006, doi:10.1007/11759966.

[4] Stec, Alexander, and Diego Klabjan. "Forecasting Crime with Deep Learning." ArXiv [Stat.ML], 5 June 2018.

[5] Felbo, Bjarke, et al. "Using Deep Learning to Predict Demographics from Mobile Phone Metadata." ICLR Workshop Track, 13 Feb. 2016.

[6] Isaksson, Emil, and Mikael Karpe Conde. "Solar Power Forecasting with Machine Learning Techniques." KTH Royal Institute of Technology, Royal Institute of Technology, 2017.