Predicting Particulate Matter 2.5 in the United States

Abstract

In this paper we train three different models: linear regression, linear regression with ridge regularization, and support vector regression on particle matter 2.5 (PM 2.5). A dangerous silent killer that is not often focused on when it comes to talking about air pollution. We found it an important substance to train on to test the predictability of it based on location data and other features. The best model, SVR with CV had the most favorable metrics which tells us that using the EPA data would be a viable way to predict the change of concentration of PM 2.5 across the US.

Introduction

Can existing air pollution data effectively predict future growth and decrease of air pollution based on location data and other features?

There is a growing problem of global climate change and air pollution. Problems such as the excess of carbon dioxide has been shown to trap heat from the Sun in Earth's atmosphere. The motivation of this project is to predict the trend of air pollution throughout the US. Since the data is specific to a latitude and longitude, there may be insights that can be extracted from the data such as which city contributes the most to air pollution and which city has had the greatest decrease according to our predictions. This has profound significance since policy makers may use this forecast to increase restrictions on pollution producing activities.

In order to maximize the predictive power of any machine learning model, it is best to use a global dataset. This makes sense since the air pollution within the U.S. isn't in a vacuum; the air travels around the world and ultimately makes its way back to the U.S. Similarly, air pollution from Europe travels around the world and eventually reaches the U.S. The stochasticity of meteorological qualities plays a considerable factor in pollution levels.[1]

Originally, we would have liked to use a global air pollution dataset, but there is no centralized global dataset that gives location specific data such as latitude and longitude. Instead, most of the data on the Internet is either aggregated by country and year or the data is fragmented by country. For example, one is able to download specific air pollution data for Netherlands that details precisely the amount of air pollution there is at a point in time at a specific location, but there is no centralized database where this information can be polled for all the countries in the world that have such data available. It would be unfeasible and outside the scope of this project to create a comprehensive web scraper and the necessary infrastructure to gather and store the data. Therefore, we decided to use the United States Environmental Protection Agency's dataset which is accessible through Google Cloud Platform (GCP) Marketplace.[2] This allowed us to easily make SQL queries on the data and leverage the computation resources of GCP BigQuery.

Background

[1] https://www.iqair.com/us/blog/air-quality/can-air-pollution-be-predicted
[2] https://console.cloud.google.com/marketplace/product/epa/historical-air-quality

Pollution prediction is not something that has never been tried before. There have been previous attempts to predict air pollution in Macau[3], Mexico City[4], and Hong Kong[5] with Support Vector Machines (SVM). Typically, SVM is used for classification, but from Smola's thesis[6] a modification in the loss function allows for this model to be used in regression as well. SVM is shown to be the dominant model to predict air pollution in various parts of the world; however, Drucker[7] and others have proposed a variant of SVM called Support Vector Regression (SVR) for regression tasks. There has been an increased interest in attempting to use this new model for predicting the air pollution such as in California[8] and Bangkok[9]. Generally speaking, SVR performs better than SVM at least for air pollution prediction. It is also known to be robust against collinear features.[10]

The research team led by Mauro Castelli trained two SVR models which included one with principal component analysis to predict the air pollution in California. They predicted air pollution for PM 2.5, Ozone, NO_2, SO_2, and CO. Since our scope only includes PM 2.5, we will be exclusively focusing on that part of their study. For data preprocessing they used imputation for missing data values and replaced them with the mode of a nominal feature and used a 2^{nd} order polynomial interpolation for numerical feature imputation. They removed outliers which were only detected for the CO measurement. For feature extraction, they extracted several features from the datetime feature. These included the month, hour of the day, season, and the boolean isWeekend. In total they had 46 features. For feature selection, they used a Pearson correlation matrix to identify dependent features and remove them. Principal component analysis was also used instead of feature selection to reduce the dimensionality and address the issue of collinearity. Random grid search was used to find the optimal parameters for SVR. They found that SVR with RBF kernel consistently outperformed other kernels, so the emphasis was on using the RBF kernel. For the PM 2.5 dataset they found an optimal C of 3 and ε of 0.032. Mauro et al. decided to use SVR with RBF kernel with PCA to obtain the best predictions but found that the non-PCA features performed slightly better with a R^2 of 0.767. The full set of metrics for both models can be found in table 1.

Methods

[3] C. M. Vong, W. F. Ip, P. K. Wong, and J. Y. Yang, "Short-term prediction of air pollution in Macau using support vector machines," Journal of Control Science and Engineering, vol. 2012, Article ID 518032, 11 pages, 2012.
[4] A. Sotomayor-Olmedo, M. A. Aceves-Fernández, E. Gorrostieta-Hurtado, C. Pedraza-Ortega, J. M. Ramos-Arreguín, and J. E. Vargas-Soto, "Forecast urban air pollution in Mexico city by using support vector machines: a kernel performance approach," International Journal of Intelligence Science, vol. 3, no. 3, pp. 126–135, 2013.
[5] W.-Z. Lu and W.-J. Wang, "Potential assessment of the "support vector machine" method in forecasting ambient air pollutant trends," Chemosphere, vol. 59, no. 5, pp. 693–701, 2005.
[6] A. Smola, C. Burges, H. Drucker et al., "Regression Estimation with Support Vector Learning Machines," Technische Universität München, München, 1996.
[7] H. Drucker, C. J. C. Burges, L. Kaufman, A. Smola, and V. Vapnik, "Support vector regression machines," Advances in Neural Information Processing Dystems, vol. 1, pp. 155–161, 1997.
[8] Mauro Castelli, Fabiana Martins Clemente, Aleš Popovič, Sara Silva, Leonardo Vanneschi, "A Machine Learning Approach to Predict Air Quality in California", Complexity, vol. 2020, Article ID 8049504, 23 pages, 2020. https://doi.org/10.1155/2020/8049504
[9] S. Arampongsanuwat and P. Meesad, "Prediction of PM 10 using support vector regression," International Conference on Information and Electronics Engineering, vol. 6, pp. 120–124, 2011.
[10] P. S. Gromski, E. Correa, A. A. Vaughan, D. C. Wedge, M. L. Turner, and R. Goodacre, "A comparison of different chemometrics approaches for the robust classification of electronic nose data," Analytical and Bioanalytical Chemistry, vol. 406, no. 29, pp. 7581–7590, 2014.

The three main methods used in this project are Linear Regression, Linear Regression with Ridge Regularization and Support Vector Regression.

The ordinary least squares linear regression or linear regression model works by minimizing the sum of residuals between the regression line and the data. This is a useful model to have since it is highly interpretable. The coefficients of the regression line show how much a certain feature influences the target variable. This adds more quality to the model by answering questions such as what features most influence the air pollution in addition to simply predicting the air pollution. It is important to note that this model only works well if the assumptions of the underlying data are true. The data is assumed to be linear and to not have any multicollinearity. Regardless of whether these assumptions are met, we found it useful to have the easy to code, low computational overhead linear regression model as a baseline for more complex models to follow.

The linear regression with ridge regularization model works the same as the linear regression model mentioned earlier except that there is now a penalty term λ (lambda) added to the objective function of the original ordinary least squares linear regression minimization problem. This regularization term lambda performs feature selection by imposing the square of the L2 norm on the coefficients. This has the effect of shrinking the coefficients of less important features to zero. Although there does not exist a closed form solution for this minimization problem, we have found it performs well with a dataset as big as three hundred million of datums.

The support vector regression (SVR) model works very similarly to support vector machines (SVM) where the problem is to find a hyperplane from all the features of the data that best separates the data by maximizing the margin. As a review, the slack is represented by ε (epsilon) which affects the error tolerance of the margin. A higher epsilon allows for a larger epsilon tube where misclassified points do not violate the margin. See figure 1 for a helpful visual. A term usually denoted as C is used within the optimization problem that controls how much we want to penalize a misclassified point that is outside the epsilon tube. Support vector machines use a kernel akin to feature expansion to map the data to a nonlinear space and find a separating hyperplane in that space. The kernel functions that scikit-learn natively supports are the linear, polynomial, radial basis, and sigmoid kernels. Since most of our experimentation led us to the higher performing radial basis kernel, we will discuss another hyperparameter specific to the radial basis kernel. The gaussian radial basis function kernel which is the transformation:

$$K(x_i, x_{i'}) = exp\left(-\gamma \sum_{j=1}^{p} \left(x_{ij} - x_{i'j}\right)^2\right)$$

This transformation has an additional hyperparameter γ (gamma) apart from the previous ones mentioned. The full details that differentiate SVR and SVM are beyond the scope of this project. In short, the constrained minimization problem in SVM introduces one slack variable for each training point while in SVR there are two slack variables for each training point. From there Karush-Kuhn-Tucker is applied to solve this modified primal problem. In summary the hyperparameters in SVR are ε, C, the kernel function, and in the case of the RBF kernel γ.

Experiments/Results

The air pollution history dataset was provided by the US EPA which is accessible through Google Cloud Platform Marketplace. The full dataset is divided into tables that contains information on various pollutants such as CO, NO_2, NO, O_3, PM 10, SO_2, and PM 2.5. We decided to focus our efforts on PM 2.5 for two reasons. Firstly, is that the full dataset would have been too cumbersome to work with since the dataset is large with respect to disk space. Secondly, PM 2.5 is known to be a silent killer, but it is not noticeable in everyday life.[11] The impact of PM 2.5 is tremendous, so we believe more focus should be concentrated on these invisible but deadly pollutants. GCP BigQuery allows us to be able to perform SQL queries on the table. From there we performed various queries on the full PM 2.5 table to download the data to our local machines as a csv file and to get a sense of the data. The earliest measurement for PM 2.5 is on March 15, 2008 and the latest measurement is on January 1, 2020 with a total of 31,953,017 datums. The maximum PM 2.5 measurement is 1248.9 $\mu g/m^3$ and the minimum PM 2.5 measurement is -10 $\mu g/m^3$. From now on all measurements of PM 2.5 are assumed to be in terms of micrograms per cubic meter. The full set of features are outlined in table 2.

We made some obvious preprocessing choices on Google's BigQuery servers before further preprocessing on our local machine. The date_gmt and time_gmt were preferred over the date_local and time_local, so we don't have to bother with differing time zones. The units_of_measure feature was dropped since all measurements of PM 2.5 are in terms of $\mu g/m^3$. The method_name feature was dropped in favor of the method_code feature since they represent the same method except one is as a human readable string and the other is as an integer. Finally, we filtered out all non-null qualifiers since we want to minimize the amount of outliers and anomalous behavior from the table. The table is ready to be shared to GCP Cloud Storage before being downloaded using the gcloud SDK. The local machine used has Python v3.9.9, scikit-learn v1.0.1, Pandas v1.35, and Dask v2021.11.2 installed. All of these are standard tools used in machine learning. Dask was used to load the sharded csv files from disk onto memory.

Once the data is loaded, date_gmt and time_gmt were parsed to create a single datetime_gmt. We decided to have two version of the data, one that was preprocessed and another that was not preprocessed to see any noticeable differences in model performance. The data that was not preprocessed simply converted the datetime_gmt feature into an integer where it represents the number of milliseconds from epoch time and the sample_measurement was dropped from the dataframe and made into its own target variable vector since that is the quantity we would like to predict. The version of the data that was preprocessed extracted the year, month, day, and hour from the datetime_gmt feature. There was no point in extracting the minute since all datums are precise to the closest hour. Then the datetime_gmt feature was dropped. A Pearson correlation matrix was created as shown in figure 2. We did not find any meaningful collinearity between two features except for longitude and state_code, but we decided this correlation must be a coincidence so neither of the two features were dropped from the dataframe.

The data was split by 66% for the training set and 33% for the testing set. Again, there are two different version of the data, the data that was normalized and data that was not normalized. The data that was normalized was through sklearn StandardScaler class which removes the mean and scales to unit variance. Even though data normalization is considered good practice before feeding it to the model, we want to observe the performance benefits of a model after normalization.[12]

As mentioned before the models used were linear regression, linear regression with ridge regularization, and SVR. 5-fold cross validation was used to tune the hyperparameters of linear regression with ridge regularization and SVR. For linear regression with ridge the different alphas tested were 0.1, 0.5, 1, 5, and

[11] https://www.bbc.com/future/article/20191113-the-toxic-killers-in-our-air-too-small-to-see
[12] https://developers.google.com/machine-learning/data-prep/transform/normalization

10 while optimizing the R^2 metric. For SVM the different kernels tested were linear, polynomial, radial basis, and sigmoid and the different epsilons tested were 0.1, 0.2, and 0.3 while optimizing for the negative MSE metric. The linear regression and linear regression with regularization models were performed on the full dataset while the SVM model was performed on a subset of the data. Since linear regression and linear regression with regularization performed well with preprocessing, we decided to only use preprocessed data for the SVM model to help further cut down on computational cost. SVM with cross validation was fed 10,000 datums for standardized data and 5,000 datums for non-standardized data to get the optimal parameters. From there we ran the SVM model on 100,000 datums with the optimal parameters to get better performance from the larger training set size.

Discussion

The metrics we chose to evaluate the models' performance are mean absolute error (MAE), root mean squared error (RMSE), mean absolute p error (MAPE), and R^2 error. A good model would have a MAE, RMSE, and MAPE close to zero and an R^2 close to one. These are all common metrics used in literature, so our models can be directly compared against them.

On initial testing of the hourly PM 2.5 dataset, we had some very unusually large error metrics, so we decided to join all the data such as the pollutants and meteorological conditions to the PM 2.5 dataset. However. we did not realize until much later that this would require predictions of the other pollutants and meteorological conditions to be able to predict future PM 2.5 values. These SQL queries can be found as "EPA Full Query Pt 1" and "EPA Full Query Pt 2". We believe that future work can leverage existing models that predict these values to the model in order to predict a future PM 2.5 values based on this enhanced feature set.

Going back to our original PM 2.5 table, we created several variations of the data to be fed to the machine learning models such as preprocessed data versus non-preprocessed data and normalized data vs non-normalized data. First, we ran this combination of data against the linear regression and linear regression with ridge regularization, see figures 3 & 4. We found that linear regression with standardized data consistently performed better with respect to all four metrics regardless of if the data was preprocessed or not. Comparing linear regression with preprocessed data and linear regression with non-preprocessed data, we found that linear regression with non-preprocessed data performed slightly better in terms of MAE, RMSE, and MAPE, but slightly worse in terms of R^2. The full table of metrics can be found in table 3. By looking at the coefficients of the linear regressor, we find that the most influential feature is longitude for both the linear regression with preprocessed data and linear regression with non-preprocessed data. For the linear regression with non-preprocessed data, poc and latitude are tied for the second most influential factors, and county_code is the the third most influential feature. For the linear regression with preprocessed data, state_code is the second most influential feature and poc and longitude are tied for third. The other linear regressors with and without ridge regularization shared these same influential features as well. We find that there is no clear advantage to preprocessing or not preprocessing the data, so we continue with evaluating the SVR model with preprocessed and normalized data from now on.

When first running the SVR model on the full dataset, we found that it would take more than a day for the model training to finish. After extensive trial and error, we found that 100,000 datums is a good size to train the SVR and 10,000 datums is a good size to train the SVR in a reasonable amount of time (under one hour). We also noticed that since we are taking a subsample of the full PM 2.5 table, we have the option of randomizing the subsample or simply querying the earliest 100,000 datums from the

table. Thus, we have a random subsample and an ordered subsample which we will consider training the SVR on. First, we ran SVR with 5-fold cross validation to get the optimal hyperparameters on 10,000 datums which optimized the negative mean squared error metric. Then we use those hyperparameters to train the SVR without CV on the larger 100,000 datums. From the linear, polynomial, radial basis, and sigmoid kernels, the cross validator found that radial basis was the optimal kernel for both the ordered and random subsample. Similarly, from the possible epsilon values of 0.1, 0.2, and 0.3 the cross validator found that the value 0.3 was the optimal epsilon for both the ordered and random subsample. There is also a hyperparameter gamma which we simply set to be the default which is one divided by the number features and the variance of the features. This choice was made to cut down on the computational complexity and have a trained model in a reasonable amount of time. To our surprise we found that the SVR with CV trained on the smaller subsample for both random and non-random performed better than SVR trained on the bigger subsample. The full comparison between different SVMs can be found in table 4 and figure 5. Ultimately the SVR with CV trained on the non-randomized 10,000 subsample performed the best. We would have thought that an SVR trained with more data would have the better metrics but from these results we can see this is not the case.

We can conclude that SVR with 5-fold cross validation trained on a chronologically ordered 100,000 subsample of the PM 2.5 table performed the best with a MAE of 0.41125, RMSE of 0.59236, MAPE of 2.42229, and R^2 of 0.64375. These metrics are comparable to that of current literature and so are good enough to use as a prediction model for predicting an increase or decrease of PM 2.5 in various locations within the US. Future work that can be done include optimizing the gamma hyperparameter when doing cross validation. We found that the sklearn library only utilized a single CPU core. After a bit a research multithreaded performance can be harnessed using sklearn's JobLib class. This plays very well with the Dask library we used to load the shard into memory from disk. As mentioned earlier, we created EPA Full Query Pt 1 & 2 which expands the feature space which we believe will better enhance the predictive power of any model that is trained on that dataset. However, in order to use the model, one must know the value of a pollutant features in advance in order to get a PM 2.5 prediction. This may be circumvented by using existing pollutant models for data imputation.

Contributions

Abner did the initial setup of the Google Cloud Product project and GitHub repository. Abner found the air pollution dataset. He wrote the project proposal and researched existing work. The EPA Full Query Pt 1 & 2 and PM 2.5 Query (SVR) SQL queries were written by him. All of the working Python code used for the project except for the support vector regression model code was written by him. He made the madness presentation slide along with a script his teammates can read from. All of the jupyter notebook execution was done on his computer since it was the fastest out of anyone's computer. He made an excel spreadsheet with the model's metrics and charts to accompany the various models. He created most of the final presentation slide deck and presented half of the slides. He edited the final presentation recording and uploaded to Canvas. Abner wrote the introduction, background, methods, experiments/results, and discussion sections of the final report. He helped teammates with accessing the dataset and answered miscellaneous pandas/scikit-learn questions.

Enoc worked on the initial query selection for the PM 2.5 data from the EPA dataset. He implemented the SVR model and 5-fold cross validation for that model. He worked on the slides for SVR on the presentation and presented the other half of the presentation. On the final paper Enoc took care of

the abstract, modifying the goal of the paper, explaining the Mauro Castelli paper, explaining the SVR model and some of the discussions about the results.

Code

https://github.com/Abner3/cs-334-project

Works Cited

A. Sotomayor-Olmedo, M. A. Aceves-Fernández, E. Gorrostieta-Hurtado, C. Pedraza-Ortega, J. M. Ramos-Arreguín, and J. E. Vargas-Soto, “Forecast urban air pollution in Mexico city by using support vector machines: a kernel performance approach,” International Journal of Intelligence Science, vol. 3, no. 3, pp. 126–135, 2013.

C. M. Vong, W. F. Ip, P. K. Wong, and J. Y. Yang, “Short-term prediction of air pollution in Macau using support vector machines,” Journal of Control Science and Engineering, vol. 2012, Article ID 518032, 11 pages, 2012.

H. Drucker, C. J. C. Burges, L. Kaufman, A. Smola, and V. Vapnik, “Support vector regression machines,” Advances in Neural Information Processing Dystems, vol. 1, pp. 155–161, 1997.

Mauro Castelli, Fabiana Martins Clemente, Aleš Popovič, Sara Silva, Leonardo Vanneschi, "A Machine Learning Approach to Predict Air Quality in California", *Complexity*, vol. 2020, Article ID 8049504, 23 pages, 2020. https://doi.org/10.1155/2020/8049504

Meng Dun, Zhicun Xu, Yan Chen, Lifeng Wu, "Short-Term Air Quality Prediction Based on Fractional Grey Linear Regression and Support Vector Machine", *Mathematical Problems in Engineering*, vol. 2020, Article ID 8914501, 13 pages, 2020. https://doi.org/10.1155/2020/8914501

S. Arampongsanuwat and P. Meesad, “Prediction of PM 10 using support vector regression,” International Conference on Information and Electronics Engineering, vol. 6, pp. 120–124, 2011.

W.-Z. Lu and W.-J. Wang, “Potential assessment of the “support vector machine” method in forecasting ambient air pollutant trends,” Chemosphere, vol. 59, no. 5, pp. 693–701, 2005.

P. S. Gromski, E. Correa, A. A. Vaughan, D. C. Wedge, M. L. Turner, and R. Goodacre, “A comparison of different chemometrics approaches for the robust classification of electronic nose data,” Analytical and Bioanalytical Chemistry, vol. 406, no. 29, pp. 7581–7590, 2014.

Appendix

Error metrics	PCA SVR-RBF		SVR-RBF	
	Training set	Validation set	Training set	Validation set
MAE	0.200	0.382	0.144	0.331
R^2	0.882	0.563	0.937	0.647
RMSE	0.273	0.576	0.205	0.512
nRMSE	0.041	0.074	0.031	0.066

Table 1: Error Metrics for PCA SVR-RBF and SVR-RBF on Training set and Validation set.

Feature Name	**Description**
state_code	The Federal Information Processing Standards (FIPS) code for the state of where the datum was obtained
county_code	The FIPS code for the county of where the datum was obtained
site_num	The unique numeric identifier of the site within the state
parameter_code	The Air Quality System (AQS) code corresponding to the parameter measurement of the system used
poc	The “Parameter Occurrence Code” is used to distinguish different instruments that use the same parameter code at the same site
latitude	The latitude of the site’s location
longitude	The longitude of the site’s location
datum	The Datum of the latitude and longitude measurements
parameter_name	The AQS name or description of the parameter measured by the monitor
date_local	The date the sample was taken on the local date of the site’s location
time_local	The 24-hour time the sample was taken on the local time of the site’s location
date_gmt	The date the sample was taken on the Greenwich Mean Time date
time_gmt	The 24-hour time the sample was taken on the Greenwich Mean Time
sample_measurement	The numeric measured value of the sample taken for the parameter
units_of_measure	The units of measurement for the sample
mdl	The “Method Detection Limit” is the minimum sample concentration the instrument can detect, if below this limit, half the MDL is reported
uncertainty	The total measure of uncertainty about the accuracy of a sample measurement
qualifier	The indicator for a missing sample or if the sample was out of ordinary (outside factor interreference such as a natural disaster)
method_type	The indicator of if the sample is a federally, regional, or other approved method
method_code	The numeric code for the method
method_name	The short description of the method
state_name	The name of the state where the monitoring site is located

county_name	The name of the county where the monitoring site is located
date_of_last_change	The last time this datum was updated within the AQS data system

Table 2: A full list of features in the PM 2.5 table.

	MAE	**RMSE**	**MAPE**	**R^2**
No Preprocessing				
LR	4.93598	7.68034	7.96228E+14	0.03292
LR with Ridge	4.93598	7.68034	7.96228E+14	0.03292
LR (std)	0.63145	0.98253	1.70542	0.03292
LR with Ridge (std)	28.56598	28.58746	233.11166	-817.70264
With Preprocessing				
LR (pp)	4.97292	7.71766	7.87784E+14	0.02350
LR (pp & std)	0.63617	0.98730	1.57616	0.02350

Table 3: The full set of metrics for the linear regression and linear regression with ridge regularization models with various kinds of data. pp is preprocessed data and std is standardized data.

	MAE	**RMSE**	**MAPE**	**R^2**
100,000 ordered subsample				
SVR (std)	0.49499	0.80762	1.58342	0.34291
SVR (std, CV)	0.41126	0.59236	2.42229	0.64375
100,000 random subsample				
SVR (std)	0.57779	0.91552	1.59337	0.09908
SVR (std, CV)	0.46844	0.88143	1.47029	0.17571

Table 4: The full set of metrics for the SVR model with various kinds of data. pp is preprocessed data, std is standardized data, and cv is cross validation.

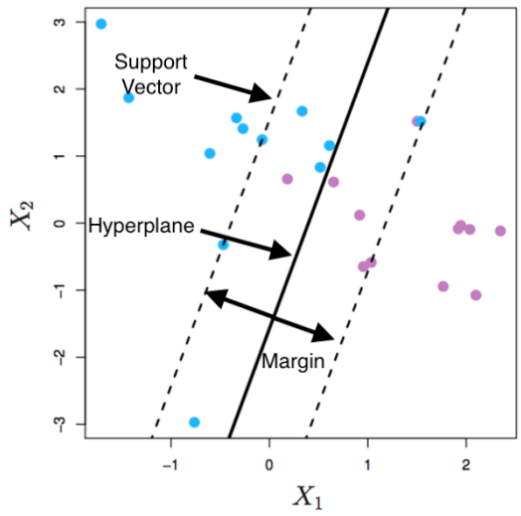

Figure 1: Illustrating the hyperplane and its components[13]

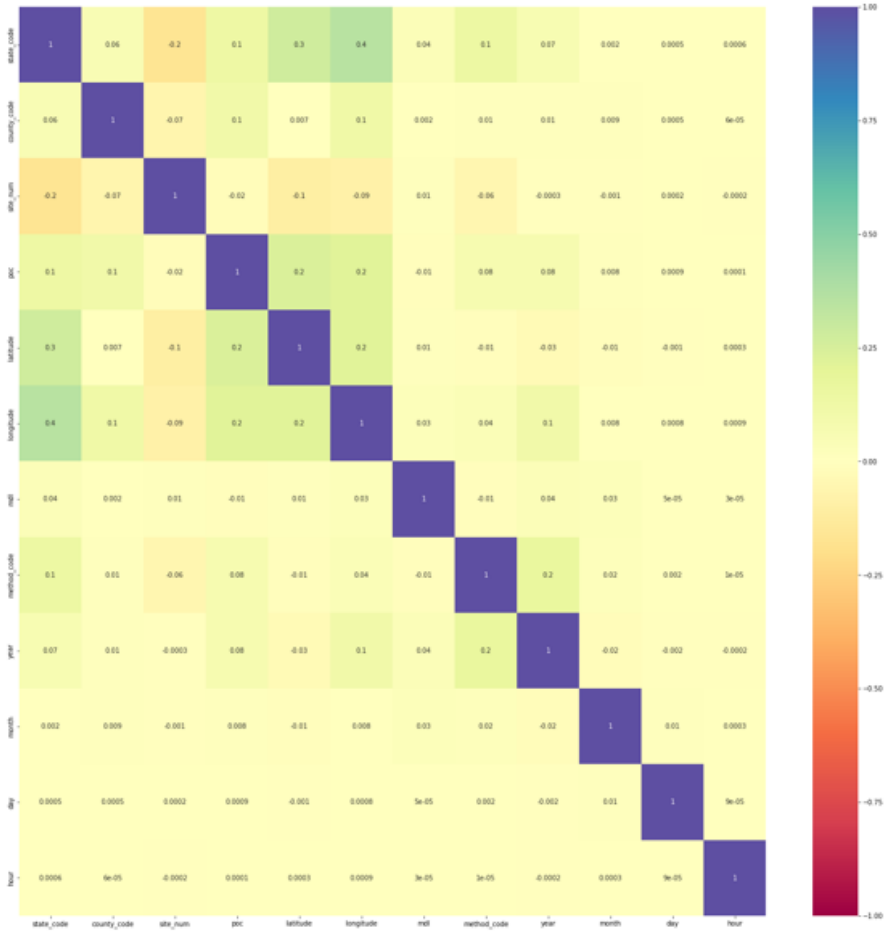

Figure 2: Pearson correlation matrix for the preprocessed data.

[13] Modified figure from CS 334 SVM lecture slides

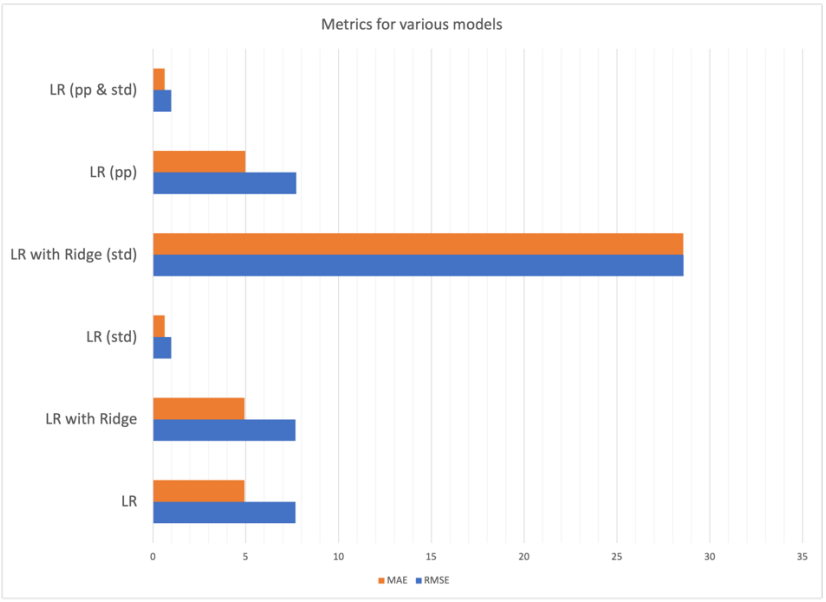

Figure 3: pp is preprocessed data and std is standardized data. MAPE was not visualized since the range is very large and would scale the x axis towards very large intervals. A bar closer to zero is better.

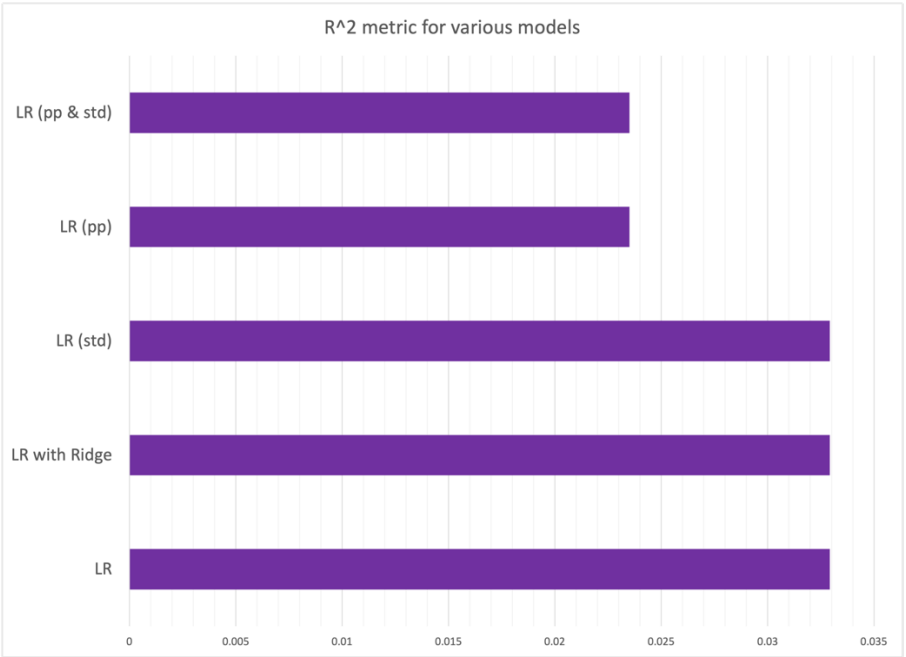

Figure 4: pp is preprocessed data and std is standardized data. A bar closer to one is better.

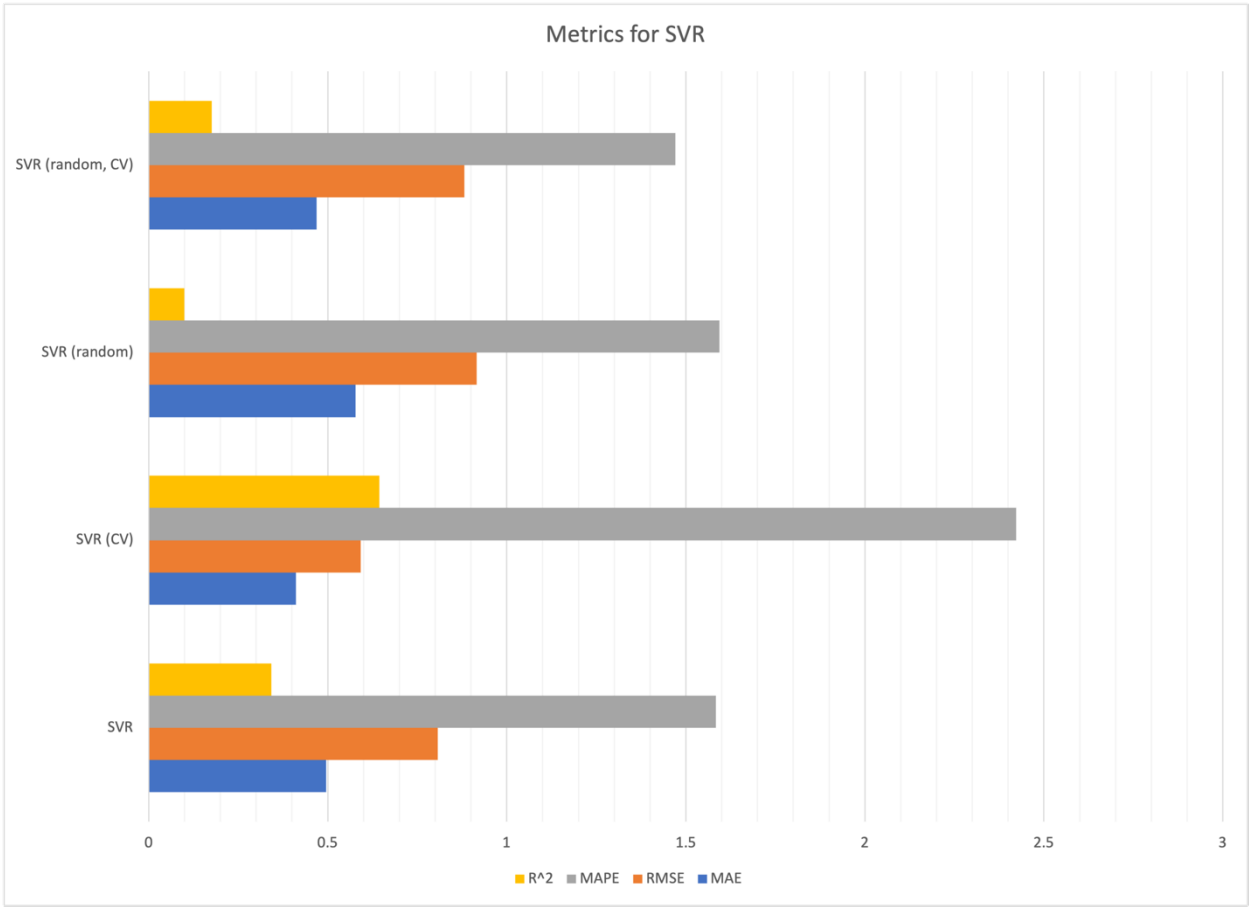

Figure 5: R^2 bars that are closer to one are better. All other bars are better when closer to zero.