1) The dataset I chose to use is from the University of California – Irvine's ML Repository

Fanaee-T, Hadi. "Bike Sharing." UCI Machine Learning Repository, 2013, https://doi.org/10.24432/C5W894.

This Dataset contains 17,389 instances of recorded data, the dataset owner provides this description of the columns:

- instant: record index

- dteday : date

- season : season (1:springer, 2:summer, 3:fall, 4:winter)

- yr : year (0: 2011, 1:2012)

- mnth : month (1 to 12)

- hr : hour (0 to 23)

- holiday : weather day is holiday or not (extracted from http://dchr.dc.gov/page/holiday-schedule)

- weekday : day of the week

- workingday : if day is neither weekend nor holiday is 1, otherwise is 0.

+ weathersit :

  - 1: Clear, Few clouds, Partly cloudy, Partly cloudy

  - 2: Mist + Cloudy, Mist + Broken clouds, Mist + Few clouds, Mist

  - 3: Light Snow, Light Rain + Thunderstorm + Scattered clouds, Light Rain + Scattered clouds

  - 4: Heavy Rain + Ice Pallets + Thunderstorm + Mist, Snow + Fog

- temp : Normalized temperature in Celsius. The values are divided to 41 (max)

- atemp: Normalized feeling temperature in Celsius. The values are divided to 50 (max)

- hum: Normalized humidity. The values are divided to 100 (max)

- windspeed: Normalized wind speed. The values are divided to 67 (max)

- casual: count of casual users

- registered: count of registered users

- cnt: count of total rental bikes including both casual and registered

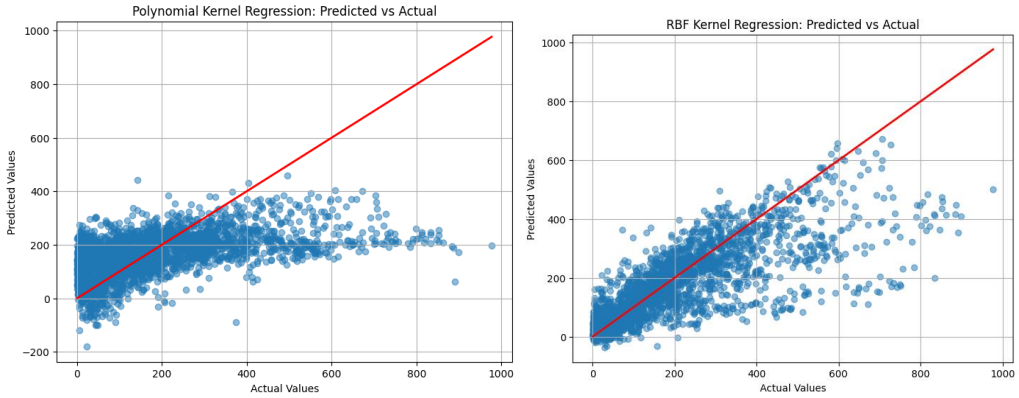
2) The task I am hoping to accomplish is predicting the “load” on the bike share system on any random given day. With a successfully trained model, one could predict the load on the bike share system in the near future.

This then could be further used to make predictions on cost if joined with other informational tables like repair costs and maintenance work.

3) Not much preprocessing was needed with this dataset as the values were already converted via standard preprocessing (numerical encoding) The steps I took with preprocessing were simply to have a clean state to return to if the programming were to fail.

4)

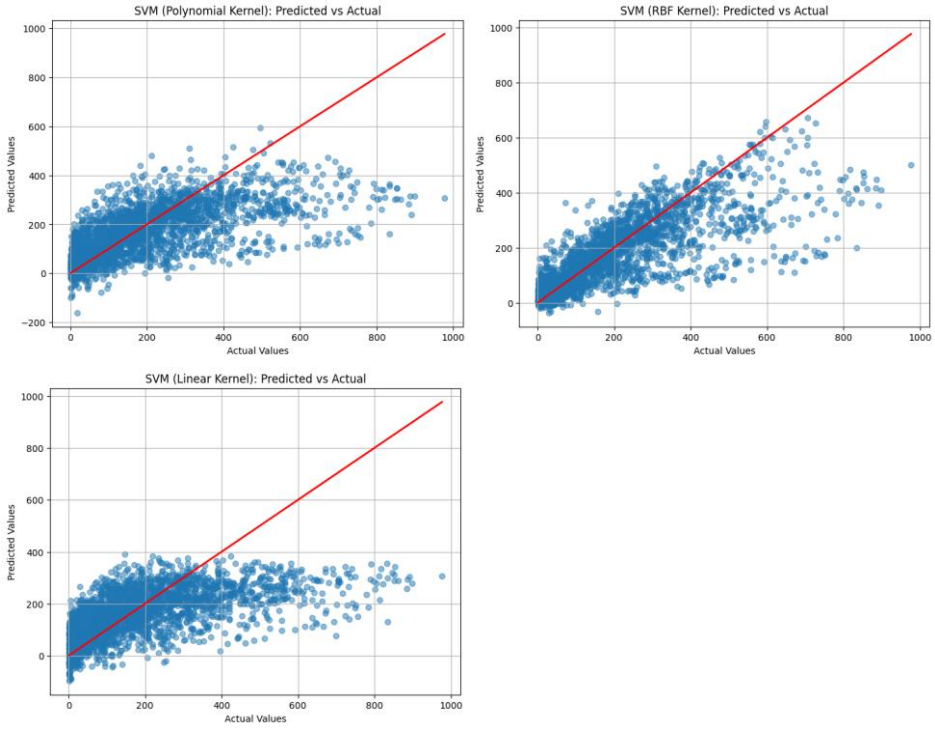
-Linear Regression



The plots above are the results of the Linear Regression models I trained, both use kernels to better attempt to fit with the red line representing a perfect fit. As you can see above the polynomial regression did not find success, with a more numerical analysis of the MSE for the Polynomial Kernel ~23094, while the RBF kernel ~12347. This is also reflected in the R^2 scores with the Polynomial Kernel ~0.25 and the RBF ~0.6, well below a “good fit” of 0.8

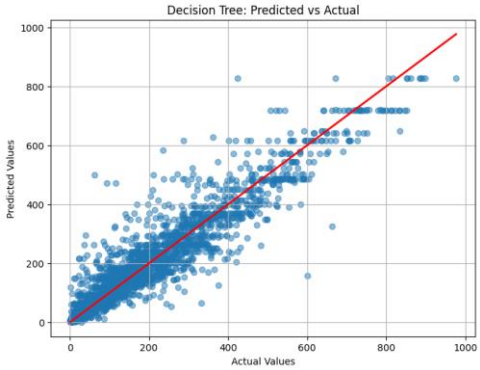
- Support Vector Machine

-



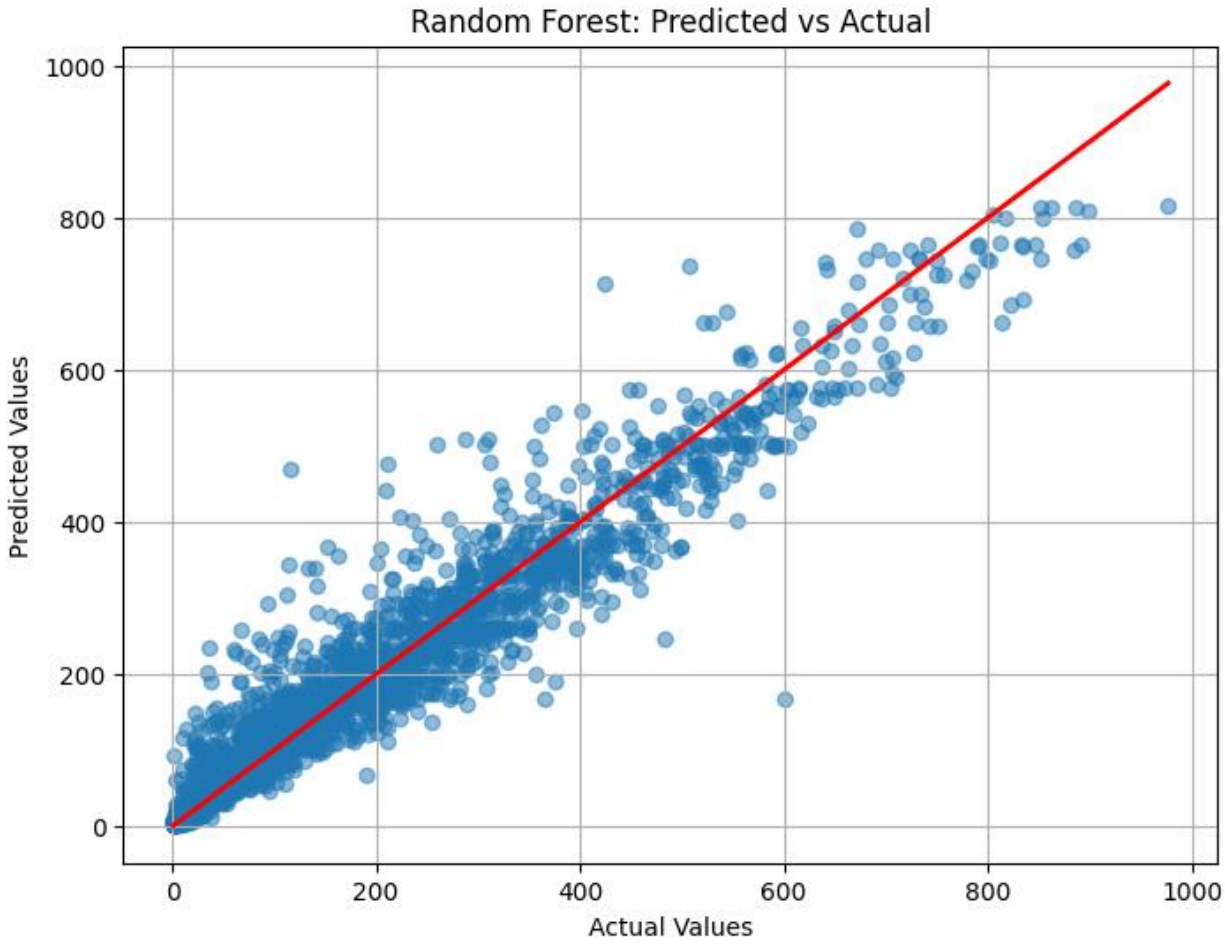
Above are the three kernel variations of the Support Vector Machine. Of the three models, the RBF Kernal saw the most success with an MSE of 12347 and R^2 of ~0.6. For this data set that is the best fit we have seen so far but there is still room for improvement

-Decision Tree



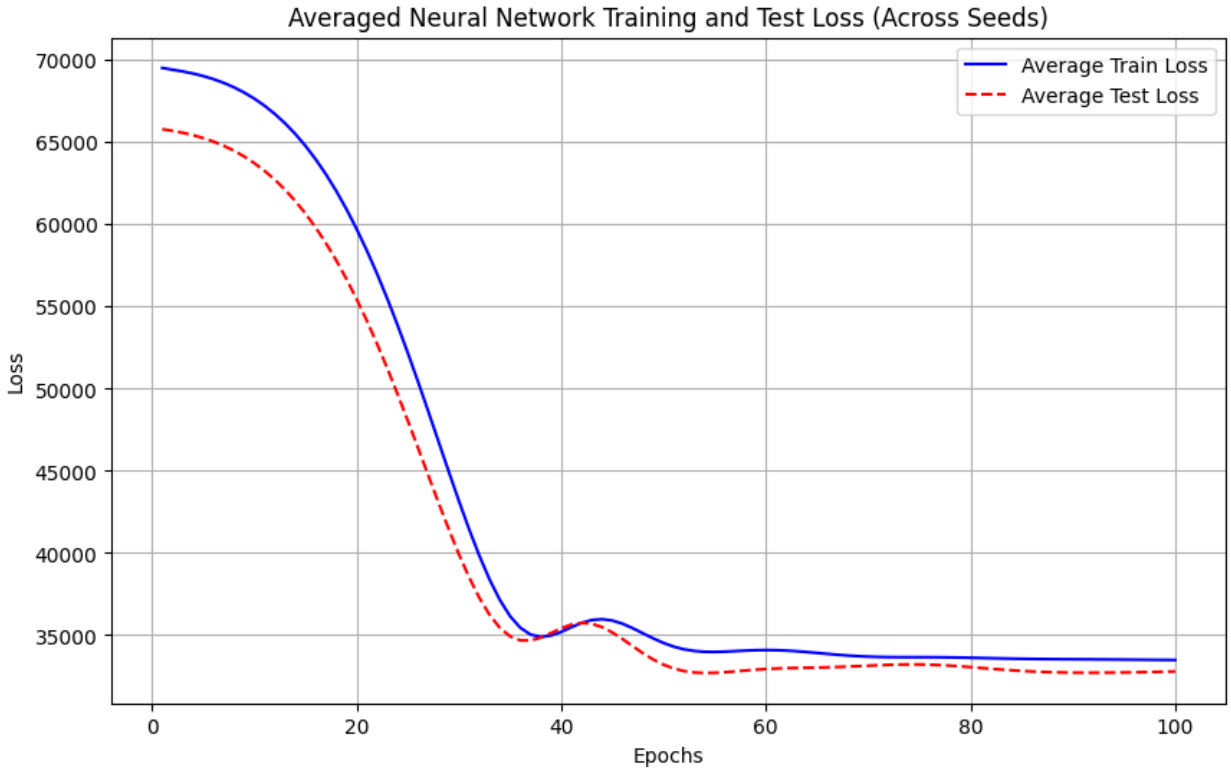
The above plot shows a single decision tree, with an MSE of 3383 and an R^2 of 0.89 I would argue that this model is a good fit with a good amount of accuracy. I suspect that this model is not over fit as we can see a number of outliers occurring the further from 0,0. This is suggestive of a model that fits well within given data but struggles to make predictions based on outliers.

- Random Forest



Based on the plot above the random forest has a good prediction accuracy, eliminating the issue with outliers a single decision tree presented, with an MSE of 2320 and an R^2 of 0.92 to reflect its success.

- Neural Network



The Neural Network allows us the advantage of seeing the Train and Test Loss over the duration of the Epochs, as the above plot shows there is convergence of accuracy around 40 epochs before the model likely falls into an overfitting territory

5.

In summary, the dataset that I have chosen is likely non-linear, as standard models like Linear Regression and SVM struggle to make accurate predictions. They do however provide a general idea on the direction of trends and could be used for quick decision making. On the other hand, decision trees and the random forest model are quick to identify feature importance and produce accurate predictions without overfitting. In a quick implementation setting a decision tree would suffice while a random forest would produce a better result. Finally, the neural network in this setting is not the best model choice, taking almost 40 epochs to converge is costly and beyond that would likely see overfitting.