# This is a report for the course INF5870.

The assignment given is to use different methods of machine learning to use in the context of wind power generation. Our goal is to get accurate predictions of generated wind power from a given wind farm, based on previously recorded data from the same farm.

A dataset TrainData.csv is given, detailing data measured from a real wind generator over 22 months. The dataset contains the following data:

* TIMESTAMP: giving date and time.
* POWER: Measured output power from wind turbine. 7
* U10: Zonal (east-west) component of forecasted wind at 10m above ground level.
* V10: Meridional (north-south) component of forecasted wind, 10m above ground level.
* WS10: Wind speed at 10m above ground level.
* U100: Zonal component of forecasted wind at 100m above ground level.
* V100: Meridional component of forecasted wind, 100m above ground level.
* WS100: Wind speed at 100m above ground level.

These data will function as our training-set for this assignment.

We then have a second dataset, WeatherForecastInput.csv, that has the same information, minus POWER, for the 13 months following TrainData.csv. We use WeatherForecasting.csv as our test-set to predict the produced power in this period.

Our third dataset Solution.csv, contains the real produced power of the wind farm in the period described in our test-set. We will try to minimize the discrepancy between our predicted output, and the actual output of the wind farm.

## Task 1

Four different approaches are being considered in this task:

* Linear regression (LR)
* K-nearest neighbor (kNN)
* Supported Vector Regression (SVR)
* Artificial neural networks (ANN)

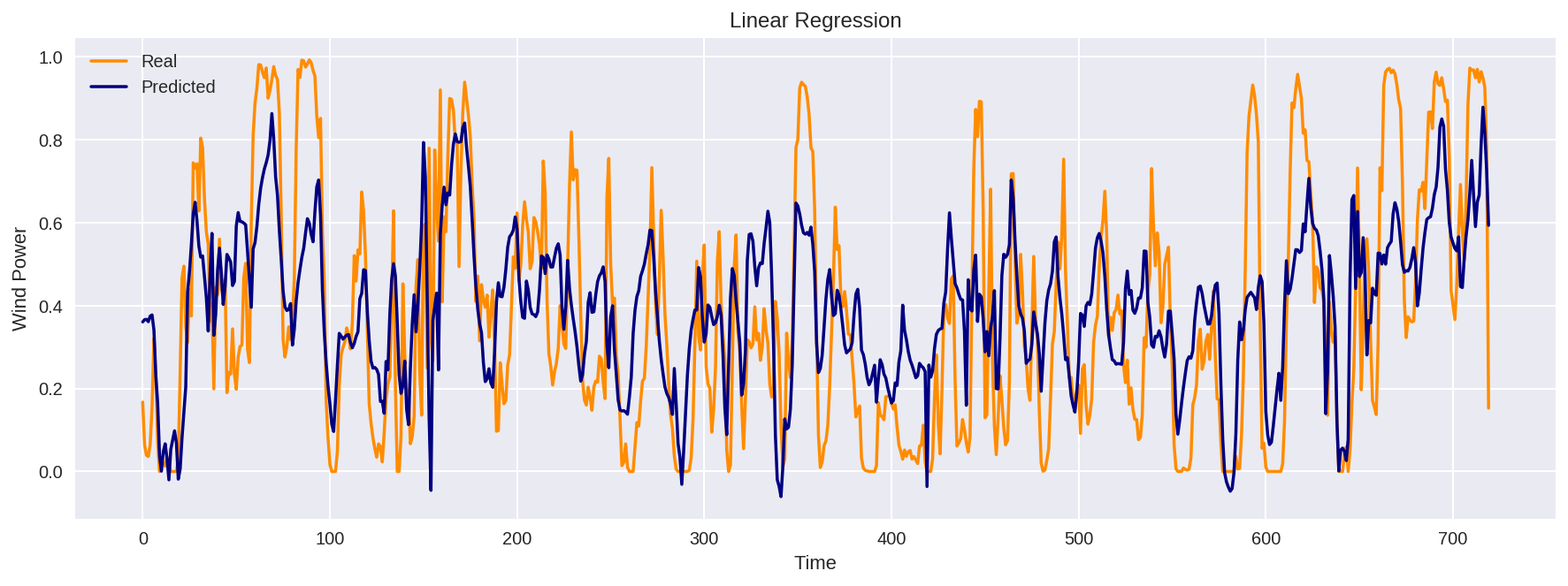
To train these models, we simply use windspeed (WS10) and output power (POWER). To figure out which one of these models produce the most accurate prediction, we check the predicted power output from each of the models against the measured power output using Root Mean Square Error (RMSE). The lower RMSE-value, the better our prediction is. We also use a coefficient of determination, R^2, as a double check on our test. The higher R^2 score, the better the prediction. A R^2 value of 1 shows a perfect prediction.

For our ANN-model, we tried out with several different numbers of hidden layers and nodes within the layers. The architecture we decided on was two hidden layers with 30 and 20 nodes, respectively, with a dropout of 20% between them.

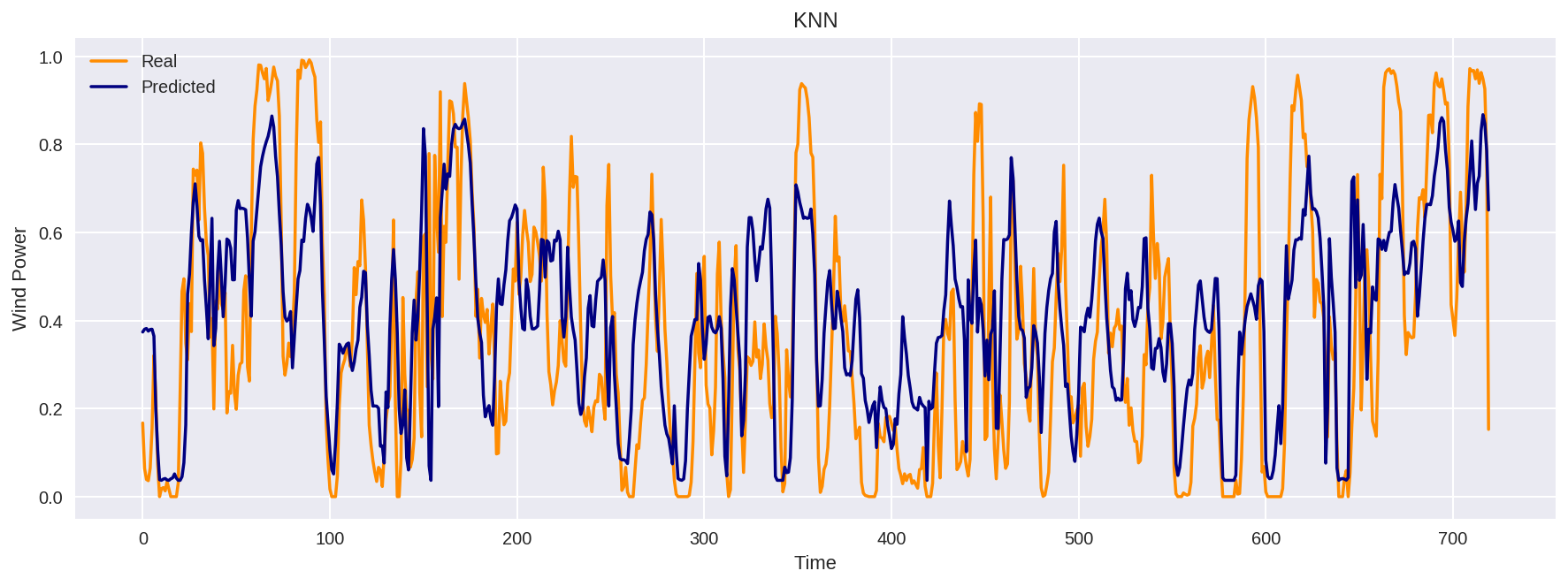
We optimized the models using grid search with cross validation, except the linear regression model as it does not contain any tweakable hyperparameters. In the case of ANN we chose to do this with trial and error instead of using cross validation grid search.

The optimal parameters found from the grid search was: KNN - 800 neighbors and uniform weights, SVR: - C=0.1, gamma=0.01

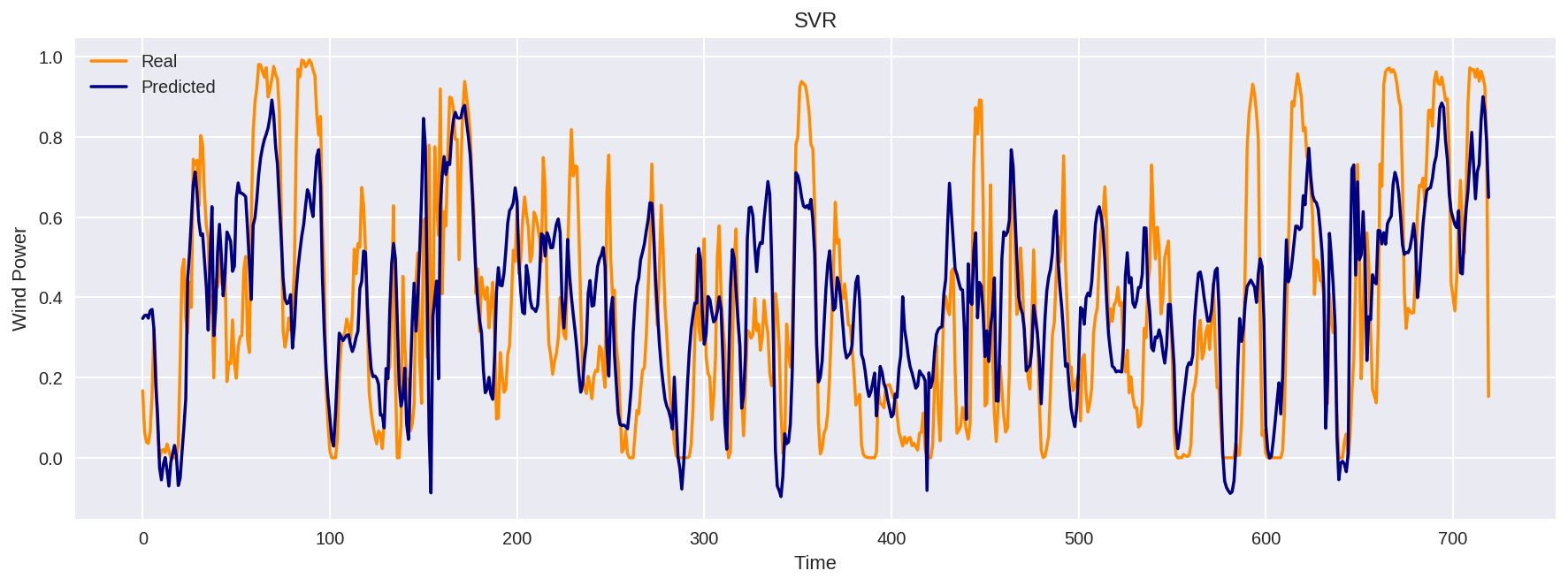
### Results:



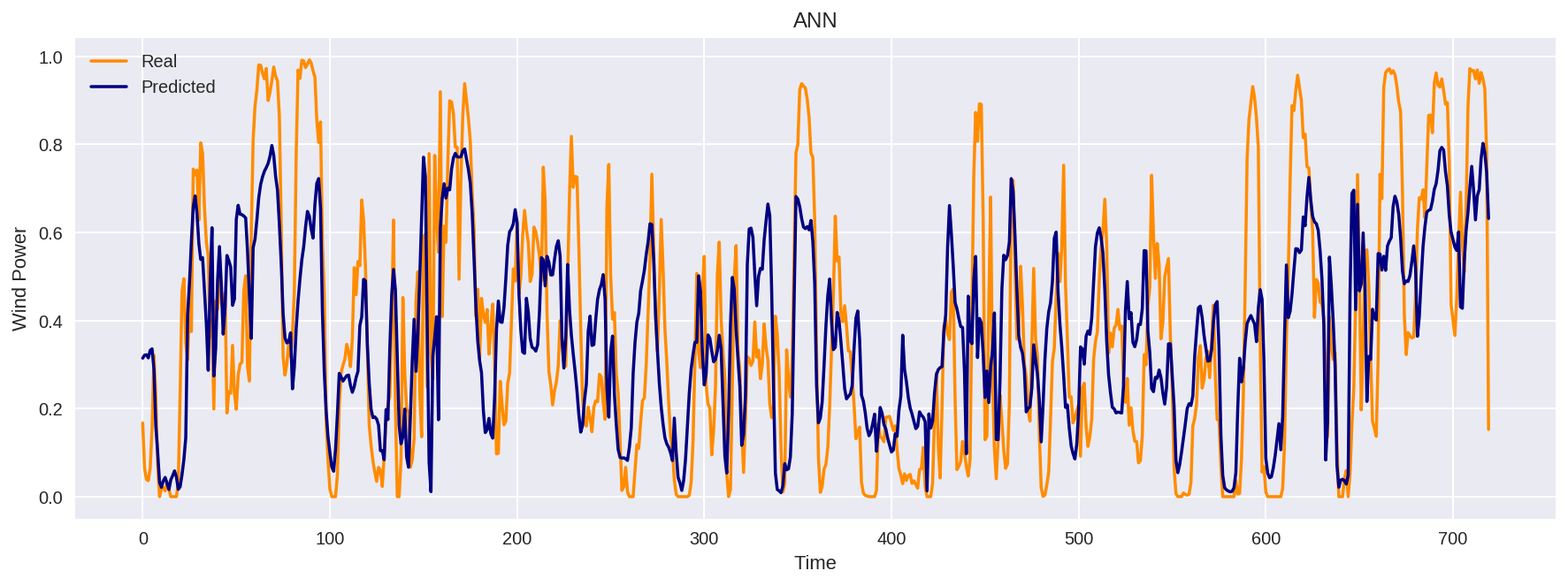
**Figure 1:** Prediction based on the linear regression model.



**Figure 2:** Prediction based on the k-nearest-neighbor model.

****

**Figure 3:** Prediction based on the supported vector regression model.

****

**Figure 4:** Prediction based on our artificial neural network model.

**Table 1: showing which models had the best prediction**

|  |  |  |
| --- | --- | --- |
| Name of model | RMSE | R2 |
| Linear regression | 0.2164 | 0.45 |
| k-nearest-neighbors | 0.2167 | 0.45 |
| Supported vector regression | 0.2158 | 0.46 |
| Artificial neural network | 0.2137 | 0.47 |

### Conclusion:

The linear regression model and the KNN model gives approximately the same result. The SVR with optimal hyperparameters gives a slighter better result, while the ANN model with manual tweaking gives the best result. All the models yield a deviation about 0.21 in production and the models explains the variance well as shown in the R2 column in table 1.

Our ANN model has a few kinks, where it produces differing quality predictions every time. For the most part it is the best model based on RMSE and R2.

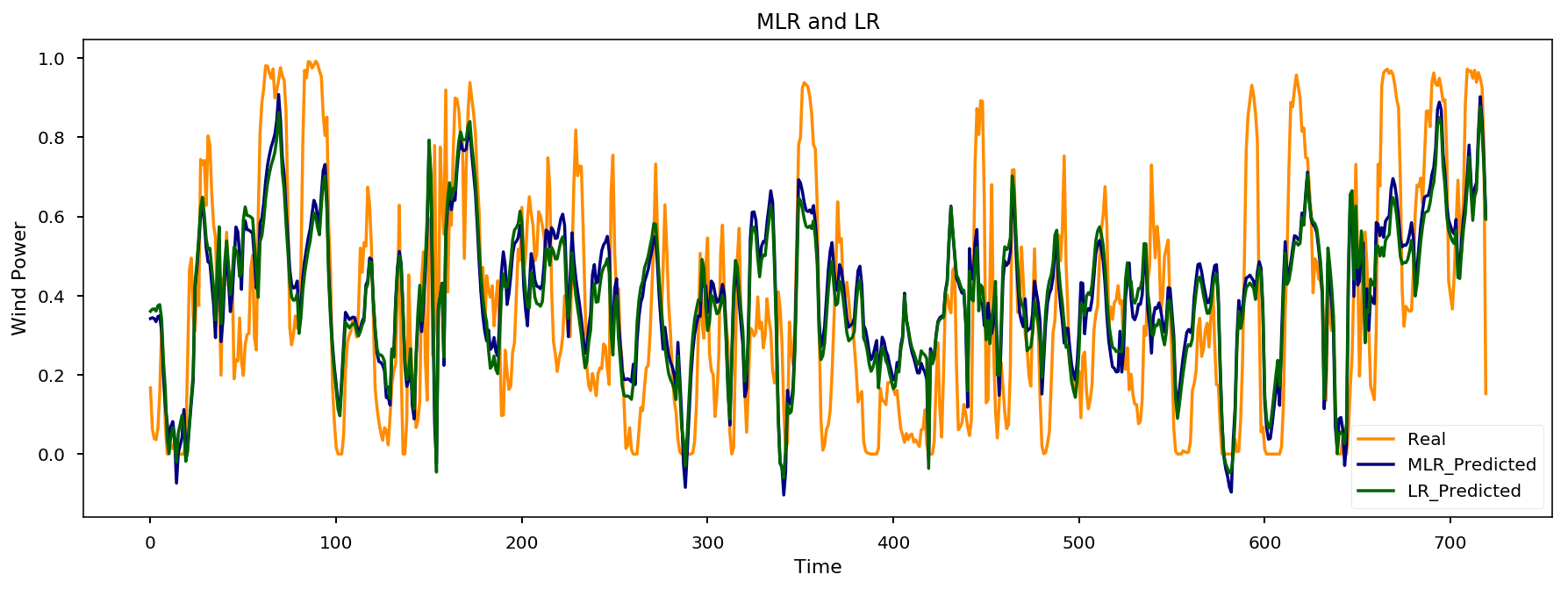
The differences in the results are due to the nature of the model algorithm. They will have their own way of finding the relationship between the wind power generation and the wind speed. The result speaks for itself.

## Task 2

In this task, we build a multiple linear regression model using both wind speed and wind direction (read from U10 and V10). This is to be contrasted with the linear regression model we used in task 1, where we only used wind speed.

Again we compare them to the real power generation data using RMSE and R^2.

### Results:



**Figure 5:** The figure shows our predictions based on a linear regression model and a multiple linear regression model.

**Table 2: Model prediction metrics of linear- and multi linear regression.**

|  |  |  |
| --- | --- | --- |
| Model name | RMSE | R2 |
| Linear regression | 0.2164 | 0.45 |
| Multiple linear regression | 0.2118 | 0.48 |

### Conclusion

The multiple linear regression model clearly makes better predictions than the linear regression model considering table 2. From figure 5 the MLR-line stretches closer to the extreme points than the LR-line does. The wind direction thusly plays a significant role in wind power production.

## Task 3

In this task we are asked to make a prediction for power generation without the use of wind speed or direction in the training data. The last thing we got to base our predictions on at this point is TIMESTAMP, date and hour.

We will use a LR model and a recurrent neural net (RNN) to solve this task.

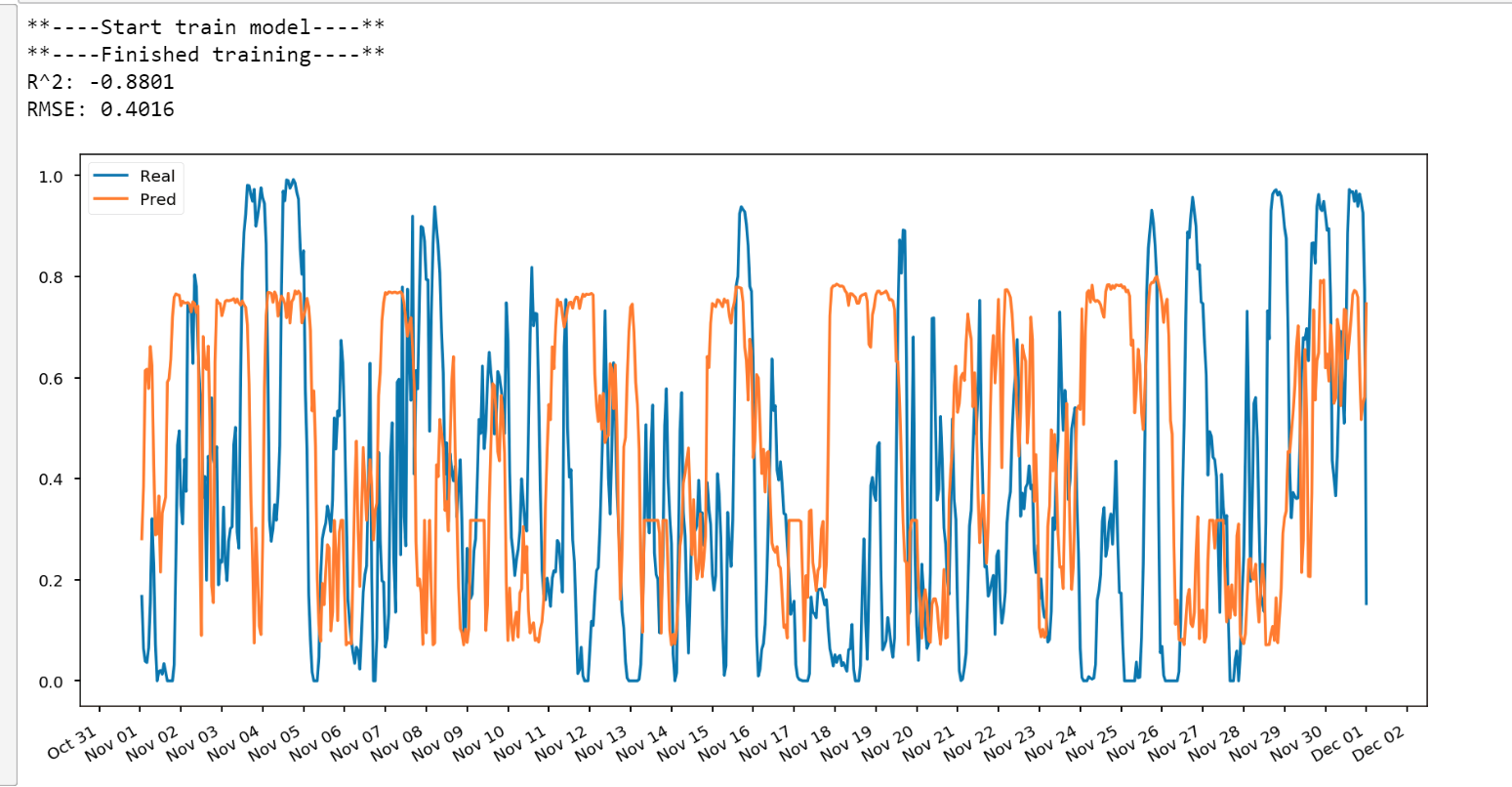
To make the training set for our RNN, we restructured the dataset so that each row contained one hour of production and the following hour of production. Our goal was to make a model where the next hour of production is predicted using the current hour’s production. The forecasted production value will then be appended to a list and used to predict the immediate next value.

To make the linear regression model, we simply removed everything from the dataset except POWER, and used index as our timestamp, since it is equivalent with one hour increments.

We used RMSE and R^2 to check which method produced the most accurate prediction.

### Results:

**Figure 6:** Prediction based on a linear regression model and recurrent neural network model using only TIMESTAMP.



**Figure 7: RNN prediction of power generation tweaking other parameters.**

**Table 3: Model prediction metrics of linear regression and recurrent neural network.**

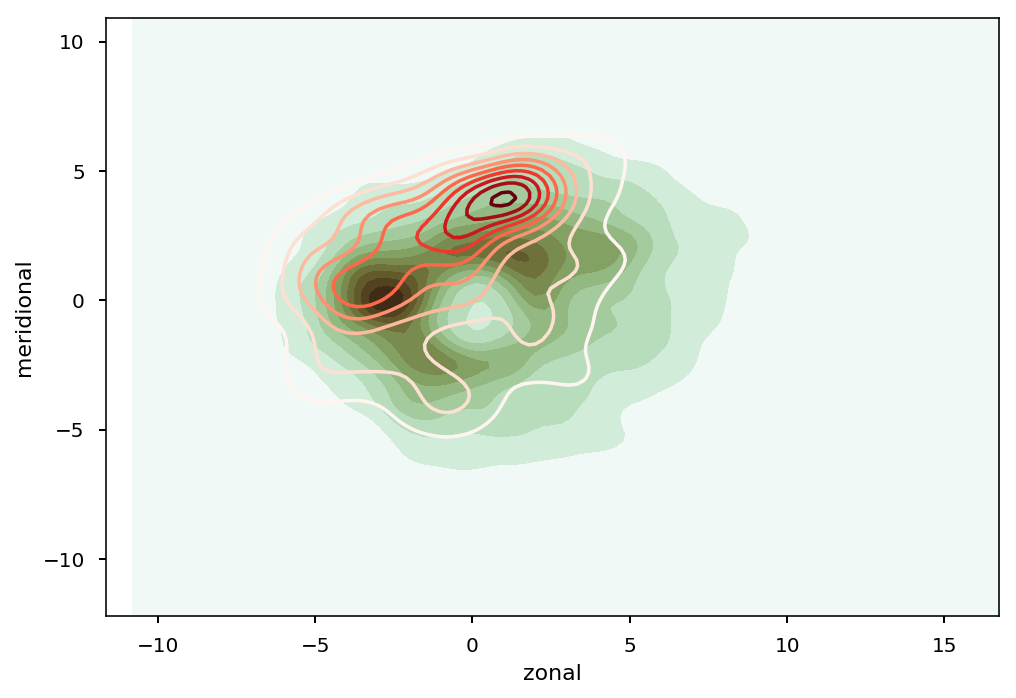
|  |  |  |
| --- | --- | --- |
| Model name | RMSE | R2 |
| Linear regression | 0.3087 | -0.11 |
| Recurrent neural network | 0.2976 | -0.0323 |

### Conclusion:

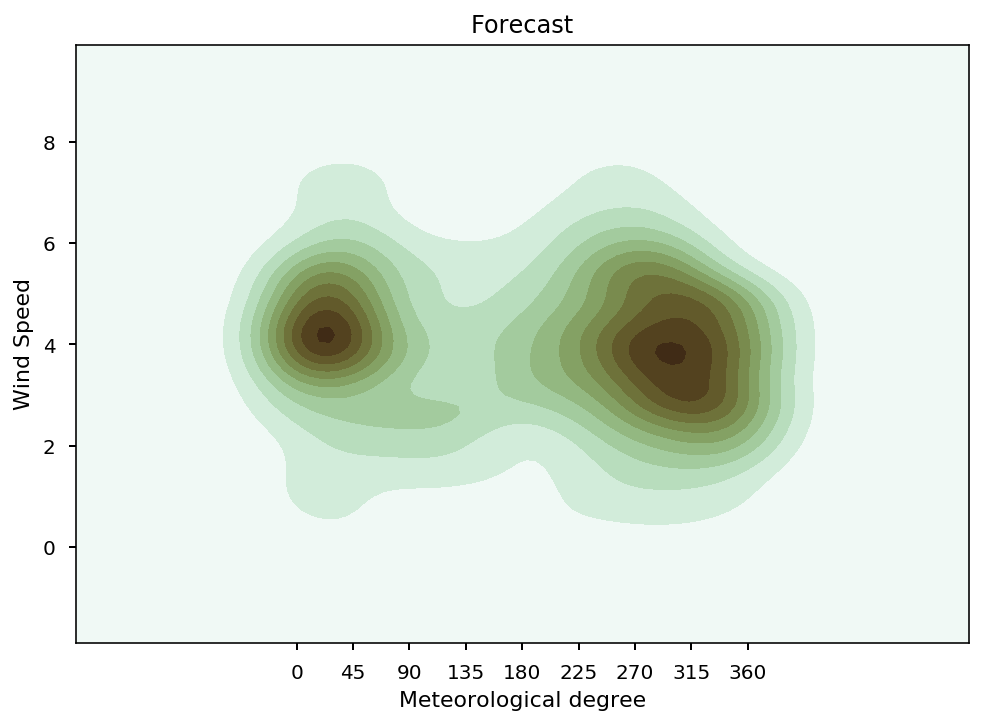
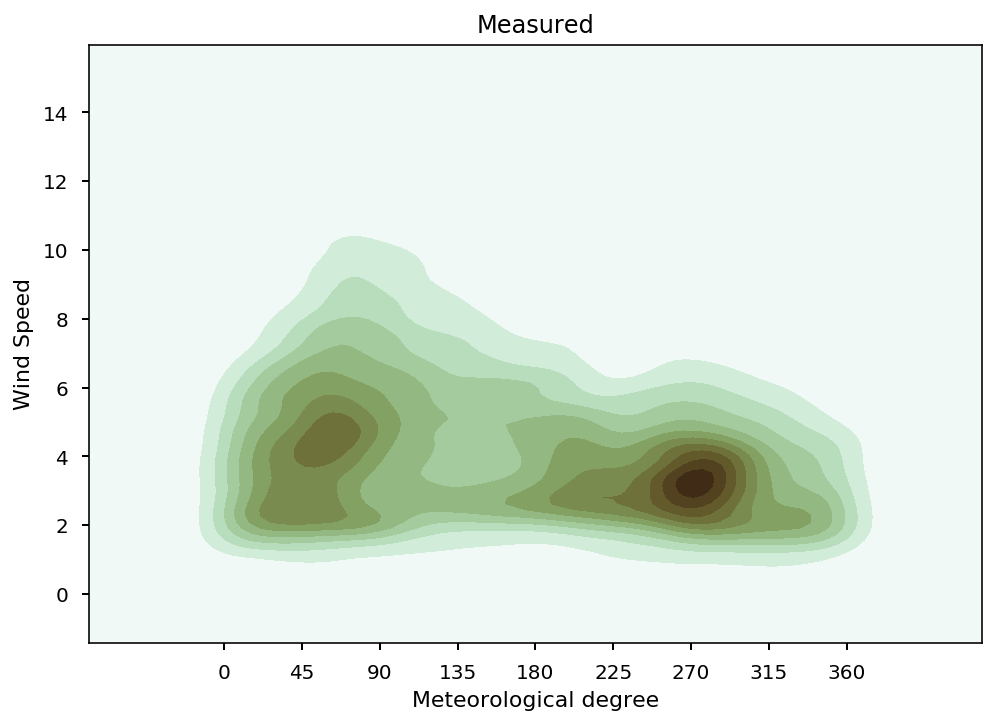
None of the models predict the generated power well at all. The best predictions result in an almost straight line. Comparing Figure 6 and 7 shows that even if the model with specific tuning with batch size of 1600 and 3000 epochs picks up variation patterns, it struggles to predict the real variations. One may end up with a deviation as high as 0.7. With a high deviation as this it may be better to use the average of the several previous power productions as a guide to forecast the next power production. The RNN model chosen suggests this method as seen in figure 6, where the line almost follows the average. As a side note, a negative R2 suggests that just taking the average of all the data would provide a better prediction than using one of our models because that would yield a lower RMSE.

## Other stuff:

Using wind directions, we created some gradient maps to visualize where winds blow from.



**Figure 8:** Red gradient shows predicted wind direction distribution. Green gradient shows measured wind direction distribution. Darker gradient means a higher rate of winds comes from the given direction.



**Figure 9:** Measured wind direction and speed. The x-axis shows angular coordinates.

**Figure 10:** Forecasted wind direction and speed. The x-axis shows angular coordinates.