Capstone Project
Machine Learning Engineer Nanodegree

Demand prediction for a bike sharing systems

July 2016
Philipp Vogler

1. Definition

1.1 Project Overview

I was looking for a problem that is solvable with machine learning in the field of transport and logistics, because this is my area of expertise. I found a very promising dataset by the company called Capital Bike Share. It is a bike sharing system in Washington DC. I use this dataset to utilizing machine learning to forecast the demand for the bike sharing system.
I applied different types of regression to find an algorithm to predict the demand for bikes based on calendric and weather information. The information about weather, calendar and bike market is available in a dataset by the University of Porto at UCI ML Repository.
This project tries to create a forecasting function based on two years of historical data by utilizing the machine learning libraries scikit-learn and tensor-flow.

1.2 Problem Statement

The long-term goal of a corporation is to make a profit. To do so, the corporation has to make decisions regarding financing and investing that factor in the current and future situation of the organization. To quantify the future position of the business, forecasts and predictions of all important performance indicators are necessary.
For a bike sharing company, the future demand for its bikes is a key indicator to consider when making decisions. Predictions about the demand are vital when scheduling maintenance of the current bicycle fleet or when to acquiring additional vehicles.
The goal of this project is to forecast the demand for bikes in dependency of weather conditions like outside temperature and calendric information e.g. holidays. This information and the demand structure is provided in a set with two years of daily historic data. The demand is given as the total daily demand and as a split for registered users and casual users. To increase the quality of the prediction

registered user demand and casual user demand will be predicted separately in step two.
To make predictions machine learning is used to train regressors. Scikit-Learn recommends a support vector regressor (SVR) for this kind of problem and data amount. Also, a deep neuronal network (DNN) regressor is trained for comparison. To find the hyper-parameters for these regressors grid search and randomized search are utilized. Due to the small dataset cross-validation is applied.

1.3 Metrics

To measure the performance of the regressions three standard regression metrics are used: Root Mean Squared Error (RMSE) and the coefficient of determination (R^2). Both metrics are calculated for both regressor types. For comparison RMSE is used and R^2 for parameter tuning. “The RMSE is directly interpretable in terms of measurement units, and so is a better measure of goodness of fit than a correlation coefficient.”

2 Analysis

2.1 Data Exploration

- instant: record index

Feature column(s):
- 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)

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

Data values:

instant	dteday	season	yr	mnth	holiday	weekday	workingday	weathersit
1	2011-01-01	1	0	1	0	6	0	2
2	2011-01-01	1	0	1	0	0	0	2
3	2011-01-03	1	0	1	0	1	1	1
4	2011-01-04	1	0	1	0	2	1	1
5	2011-01-05	1	0	1	0	3	1	1

instant	weathersit	temp	atemp	hum	windspeed	casual	registered	cnt
1	2	0.344	0.363	0.805	0.160	331	654	985
2	2	0.363	0.353	0.696	0.248	131	670	801
3	1	0.196	0.189	0.437	0.248	120	1229	1349
4	1	0.200	0.212	0.590	0.160	108	1454	1562
5	1	0.226	0.229	0.436	0.186	82	1518	1600

Data stats:

	instant	season	yr	mnth	holiday	weekday	workingday	weathersit
count	731	731	731	731	731	731	731	731
mean	366	2.49	0.50	6.51	0.02	2.99	0.68	1.39
std	211	1.11	0.50	3.45	0.16	2.00	0.46	0.54
min	1.00	1.00	0.00	1.00	0.00	0.00	0.00	1.00
25%	183.5	2.00	0.00	4.00	0.00	1.00	0.00	1.00
50%	366	3.00	1.00	7.00	0.00	3.00	1.00	1.00
75%	548	3.00	1.00	10.0	0.00	5.00	1.00	2.00
max	731	4.00	1.00	12.0	1.00	6.00	1.00	3.00

	temp	atemp	hum	windspeed	casual	registered	cnt
count	731	731	731	731	731	731	731
mean	0.49	0.47	0.62	0.19	848	3656	4504
std	0.18	0.16	0.14	0.07	686	1560	1937
min	0.05	0.07	0.00	0.02	2.00	20.0	22.0
25%	0.33	0.33	0.52	0.13	315	2497	3152
50%	0.49	0.48	0.62	0.18	713	3662	4548
75%	0.65	0.60	0.73	0.23	1096	4776	5956
max	0.86	0.84	0.97	0.50	3410	6946	8714

Characteristics

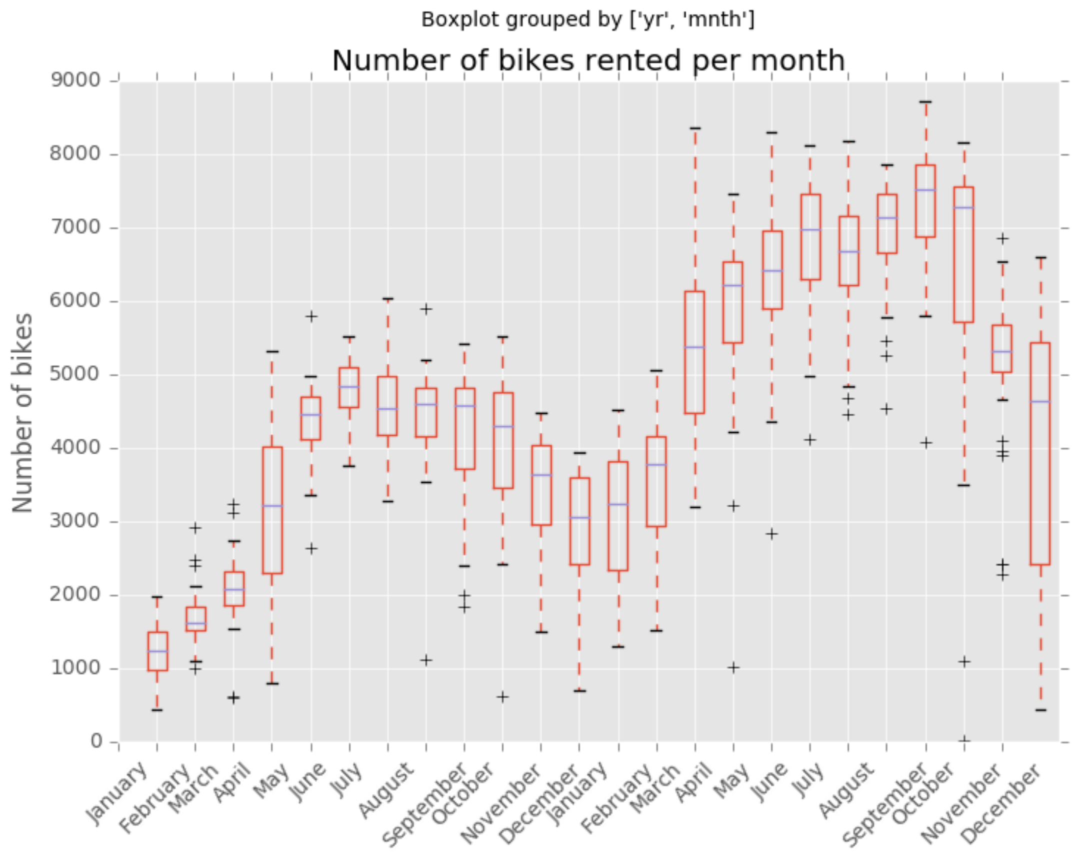
The characteristics of the dataset are very favorable because it was already processed. It is very concise, and missing values are not a problem. Also, most of the data is already normalized or binary. Other categorical data like 'weekday' or 'working day'/'holiday' were processed and transformed into dummy variables.

2.2 Exploratory Visualization

The visualization shows a classic seasonal pattern with an up-trend year over year.

Unsurprisingly bike renting in Washington DC is much more popular in the summer month. Spring and autumn months show higher volatility than the rest of the year, which is likely due to changing weather conditions.

There are some outliers throughout the dataset, mostly on the lower end. These are left in the dataset because they are not due to measurement errors, but to extreme weather conditions. Because extreme weather conditions are part of the problem the data is not excluded.



2.3 Data Preprocessing (Methodology)

As described in ‘Characteristics’ most of the preprocessing is provided with the data set. Dates get dropped because the regressor can not read this datatype and the order information is already stored in the index. The instant variable replicates this information also. These features are dropped because the order should not differentiate the data points. The January 1st of 2011 is not better or worse than January 1st 2012 by the order of the data set. It should differentiate on the years feature, but that information is stored in the ‘yr’ feature already. Keeping date (and instance) in would overemphasize these features.

2.4 Algorithms and Techniques

Creating a model to predict demand from historical data is a supervised learning process. Forecasting the rental bike demand is obviously a quantitative prediction. Therefore a regression algorithm is needed. A simple SGD (Stochastic Gradient Descent) regressor would be the first choice, but due to the limited number of data points, a more "sophisticated" regressor is needed. An SVR (Support Vector Regressor) with more eighth a linear or an RBF (radial basis function) kernel is a good choice. They operate well even if only a limited amount of data (<

10k) is available.
To see if a deep neuronal network (DNN) might be able to beat a linear model for this particular prediction problem the tensor-flow DNN Regressor algorithm is employed.

2.5 Benchmark

Two types of regressors are trained: an SVR and a DNN-Regressor. Both are first used off-the-shelf with default parameters to create a benchmark. Both "benchmarks" for the coefficient of determination are close to zero. The RMSE is pretty big and similar for both algorithms. Therefore parameter tuning is mandatory for SVR and the DNN Regressor.

Score SVR: -0.029012
RMSE SVR: 1895.485129

Score DNN: 0.031714
RMSE DNN: 1838.704961

3 Methodology

3.1 Implementation

The regressors are trained using randomized search and cross-validation to identify the area of the best parameters. Then a grid search is used to tune parameter values of the regressor functions.

SVR tuned with GridSearch and RandomizesSearch

```
Score SVR: -0.029012
Score SVR tuned GS: 0.786930
Score SVR tuned RS: 0.791948

RMSE SVR: 1895.485129
RMSE SVR tuned GS: 862.524552
RMSE SVR tuned RS: 852.306407
```

The tuning works for the SVR.

DNN-Regressor tuned with GridSearch and RandomizesSearch

```
Score DNN: 0.031714
Score DNN tuned GS: 0.131557
Score DNN tuned RS: 0.165697

RMSE DNN: 1838.704961
RMSE DNN tuned GS: 1741.328626
RMSE DNN tuned RS: 1706.758115
```

Same picture with the DNN Regressor. The tuning helps, but the results are still underwhelming. Also, the best DNN result is no match for the tuned SVR.

Results R^2

```
SVR tuned RS: 0.791948
DNN tuned RS: 0.165697
```

SVR works much better than the DNN Regressor in comparison.

3.2 Refinement

The count of rented bikes (cnt) is just the sum of the features casual and registered. Two separate models are trained to predict these features. And add up afterward. This split should improve the projection.

SVR with GridSearch - for casual users

```
Best parameter from grid search: {'kernel': 'linear', 'C': 1000}
```

SVR with RandomizesSearch - for casual users

```
Best CV score from random search: 0.633508
```

SVR with GridSearch - for registered users

Best parameter from grid search: `{'kernel': 'linear', 'C': 3000}`

SVR with RandomizesSearch - for registered users

```
Best CV score from random  search: 0.794217
```

4 Results

4.1 Model Evaluation and Validation

```
Score cas: 0.634344
Score reg: 0.799791
Score sum: 0.792683

RMSE cas: 427.873530
RMSE reg: 699.776259
RMSE sum: 850.800803
```

The coefficient of determination shows that the model is better in capturing the seasonal and trend effects with the registered users than with the casual users. Unfortunately, it is not as good in predicting the actual numbers of registered users compared to casual users (higher RMSE), because the registered users are the majority of all users. The improvements over the SVR model that predicts the data set as a whole are marginal:

```
Score SVR tuned RS: 0.791948
Score sum: 0.792683

RMSE SVR tuned RS: 852.306407
RMSE sum: 850.800803
```

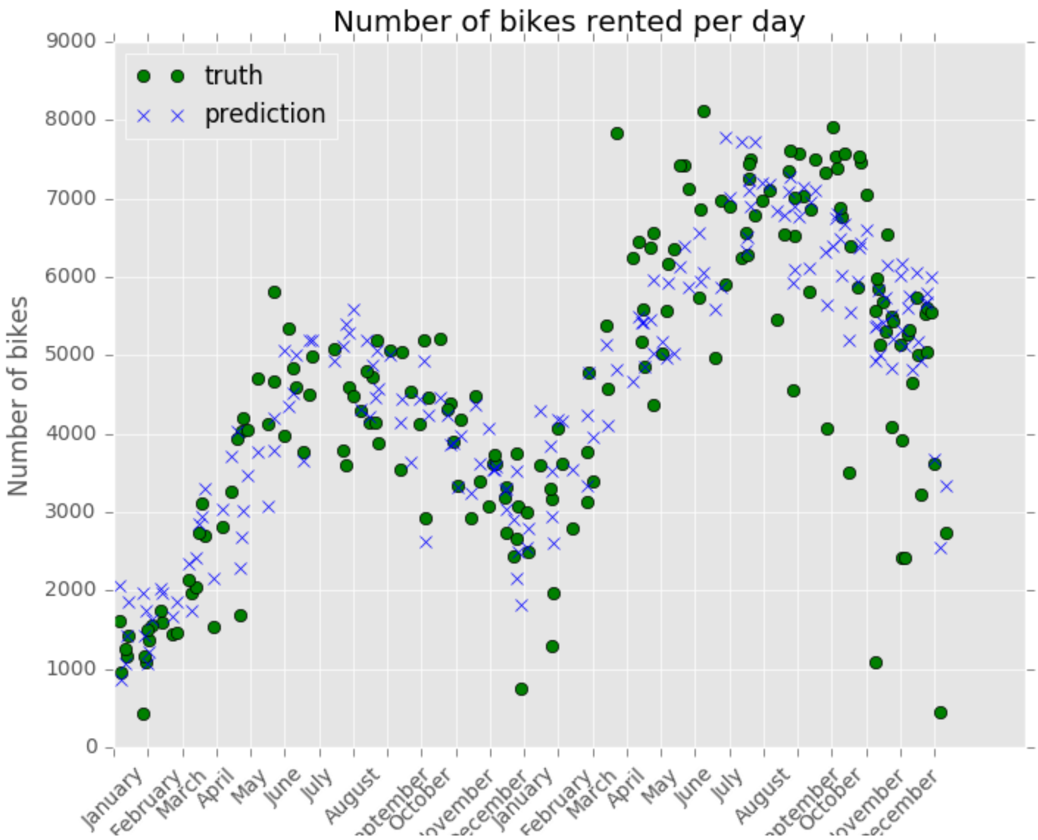
4.2 Justification

As expected the tuning of the parameters of the regressors improves the performance. Both regressors make much better predictions after tuning. Parameter tuning with random search can improve the performance even further after the right interval was identified by grid search.
Despite the tuning, the SVR beats the DNN Regressor by far. The predictions by the SVR are more than five times as good as the DNN Regressors.
Splitting the dataset and predicting casual and registered customers separately increase the R^2 score also slightly.
A coefficient of determination of more than 80% is a decent result for the SVR regressor. The DNN Regressor predictions are disappointing.

Maybe the size of the dataset is insufficient to train the DNN Regressor properly.

5 Conclusion

5.1 Free-Form Visualization



The predictions on the test set are reasonably good, without overfitting. The model captures the upward trend as well as the seasonal curve. About half of the predictions are higher than the truth, and half are lower. As expected from an SVR model, high and low outliers are not captured. The model creates balanced predictions over all.

5.2 Reflection

I had high hopes for the DNN Regressor. It was kind of disappointing that it does not even come close to the SVR results. Maybe my tuning was not right, or it needs a larger dataset or computational power. Utilizing grid and randomize search in a way that makes sense was a little tricky. At first, the results of the parameter tuning seemed random. This behavior might be caused by local minima in the solution space. Parameter tuning results became more stable when I switched the order of the search methods . It makes more sense to start with a broad grid search and then use randomized search on the given interval, instead of visa verse. It is also computational more efficient.

5.3 Improvement

The coefficient of determination of the regressors could be increased by additional iterations in training and the number of folds in the cross-validation, at the expense of computing time. More iterations might be particularly useful with the performance of DNN Regressor.
The performance of the DNN Regressor might also increase with the amount of data available. Of course, there are also other regressors available that might perform better on this particular dataset. For example, a wide and deep learning algorithm might be a better-performing alternative.

Reference

> http://www.capitalbikeshare.com
> http://archive.ics.uci.edu/ml/datasets/Bike+Sharing+Dataset
> http://freemeteo.de/wetter/
> http://dchr.dc.gov/page/holiday-schedules
> http://scikit-learn.org/stable/
> https://www.tensorflow.org
> https://www.tensorflow.org/versions/r0.9/tutorials/linear/overview.html#large-scale-linear-models-with-tensorflow
> https://www.tensorflow.org/versions/r0.9/api_docs/python/contrib.learn.html#DNNRegressor
> http://scikit-learn.org/stable/tutorial/machine_learning_map/index.html
> http://scikit-learn.org/stable/modules/generated/sklearn.svm.SVR.html#sklearn.svm.SVR
> http://scikit-learn.org/stable/auto_examples/svm/plot_rbf_parameters.html
> http://scikit-learn.org/stable/modules/sgd.html#regression
> http://www.vernier.com/til/1014/
> https://github.com/tensorflow/skflow/blob/master/g3doc/api_docs/python/estimators.md
> http://scikit-learn.org/stable/modules/generated/sklearn.grid_search.GridSearchCV.html
> http://scikit-learn.org/stable/modules/generated/sklearn.grid_search.RandomizedSearchCV.html
> http://scikit-learn.org/stable/modules/generated/sklearn.metrics.mean_squared_error.html#sklearn.metrics.mean_squared_error
> http://scikit-learn.org/stable/modules/generated/sklearn.metrics.r2_score.html#sklearn.metrics.r2_score