Machine Learning E1 Regression Task

Group 21

Sophie Rain (01425316),

Peter Stroppa (01326468),

Lucas Unterberger (01325438)

General Overview

- Used Datasets:
  - BikeSharing:
  - https://inclass.kaggle.com/c/184702-tu-ml-ws-19-bikesharing
  - Students: https://inclass.kaggle.com/c/184702-tu-ml-ws-19-student-performance
  - Airquality:
  - https://archive.ics.uci.edu/ml/datasets/Air+quality
  - Energy:
  - https://archive.ics.uci.edu/ml/datasets/energy+efficiency

General Overview

- Regression Methods (using sklearn in python):
  - Regression Tree (min_samples_leaf, max_depth)
  - Linear Regression (fit_intercept=True)
  - Lasso Regression (alpha)
  - kNN (k, weights, algorithm)

General Overview

- Preprocessing Methods
  - OneHotEncoding
  - OrdinalEncoding
  - MinMax Scaling
  - Z-score Scaling
  - Feature Selection

- Evaluation Measures
  - Rooted mean squared error
  - Relative mean squared error
  - Mean absolute error
  - Relative absolute error
  - Correlation

- No fixed Train-test-split
- random sampling in every iteration (80:20)
- Preprocessing usefull for every dataset

Comparison of Datasets

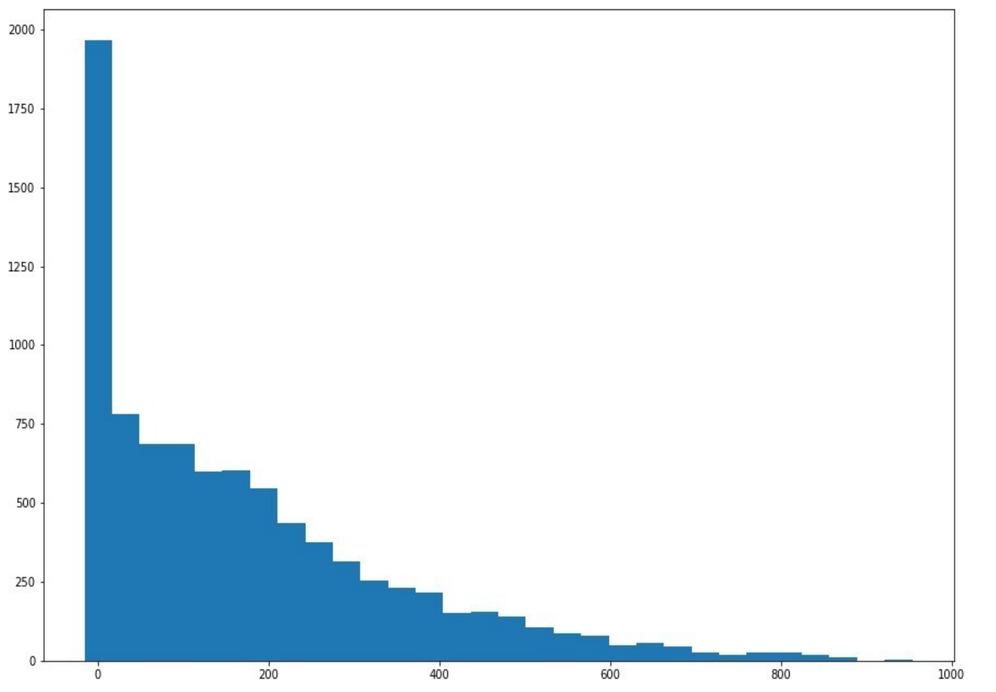
- Bike and Students datasets contain numerical and categorical data
- Airquality and Energy contain numerical values
- Airquality contains missing values
  - set to -200 by default
- Number of features: 10 (air) - 30 (students)
- Highest number of samples 9357 (air)
- Lowest number of samples 198 (students)

BikeSharing Dataset

- Number of training samples: 8690
- 15 features
  - Dteday only string value --> redundant
  - Id omitted (irrelevant)
  - Others either float or integer/ordinal valued
  - Range varies: (Preprocessing!)
    - Hour: 0-23 <-> weather information: 0-1 <-> holiday: boolean

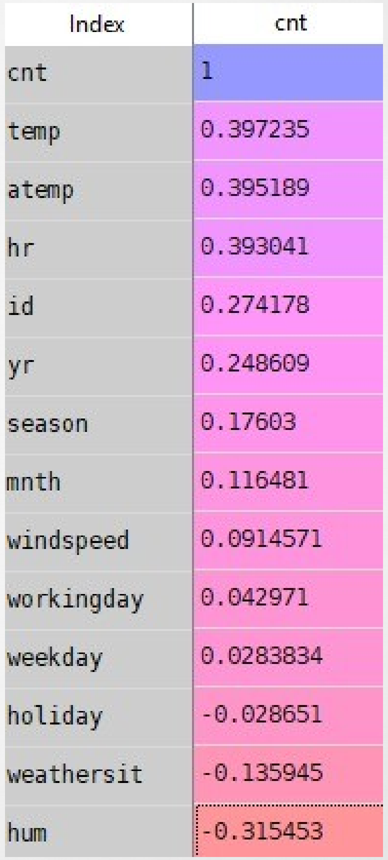
BikeSharing Dataset

- Target feature: cnt = number of rented bikes
    - Range about 0-900 (mean: 188, median 142, std: 178)
- Missing values? – No
- Frequency in graphic



BikeSharing Preprocessing

- Correlation Features <-> Target:
  - No feature with very high correlation
  - Weekday and holiday low values
  - Workingday and holiday complementary
  - → overrepresentation?



Index	cnt
cnt	1
temp	0.397235
atemp	0.395189
hr	0.393041
id	0.274178
yr	0.248609
season	0.17603
mnth	0.116481
windspeed	0.0914571
workingday	0.042971
weekday	0.0283834
holiday	-0.028651
weathersit	-0.135945
hum	-0.315453

BikeSharing Preprocessing

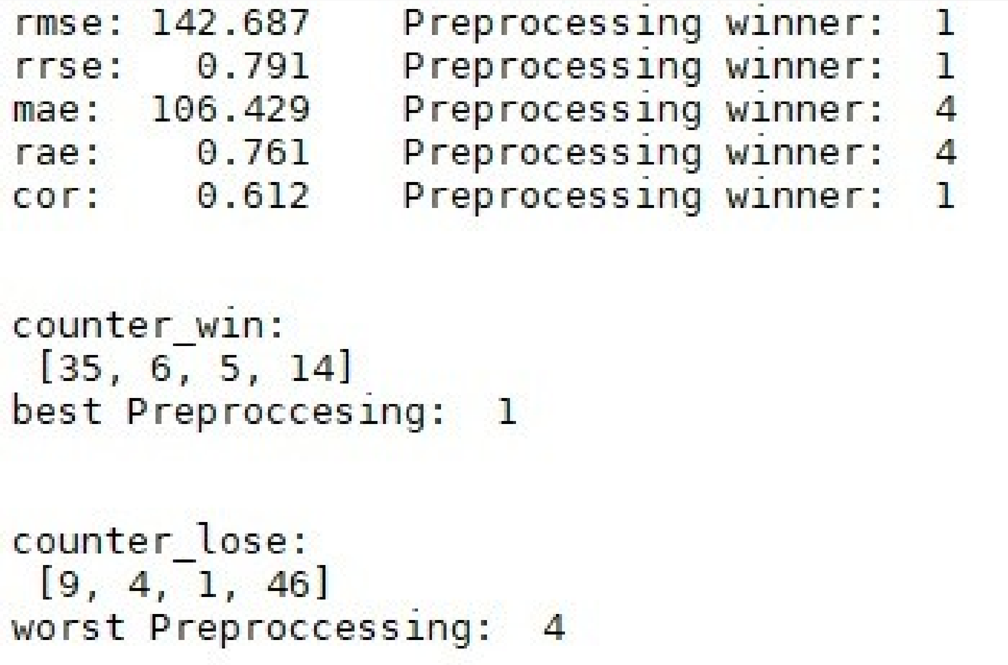
- Considered 4 different approaches:
  1. No preprocessing (other than dropping both id and dteday)
  2. Z-score scaling on all parameters
  3. Min-max scaling on all parameters
  4. Feature Selection acc. to correlation vector, i.e. drop weekday, holiday
- Results
  - For every method but kNN: Feature selection performed poorly
  - Other 3 approaches: strong dependence on method

BikeSharing - Linear Regression

- Preprocessing:
  - The raw data approach yields best result
  - Worst result: Feature Selection
- Results-Preprocessing:
  - 20 runs, different train-test splits
  - Every time evaluation of 5 measures (see graphic), 2 redundant
  - Saved best and worst preprocessing and counted (3*20 counts)

BikeSharing - Linear Regression

- Measures:
  - Rooted mean squared error: 143
  - relative error: 79%
  - Not good, at least <std



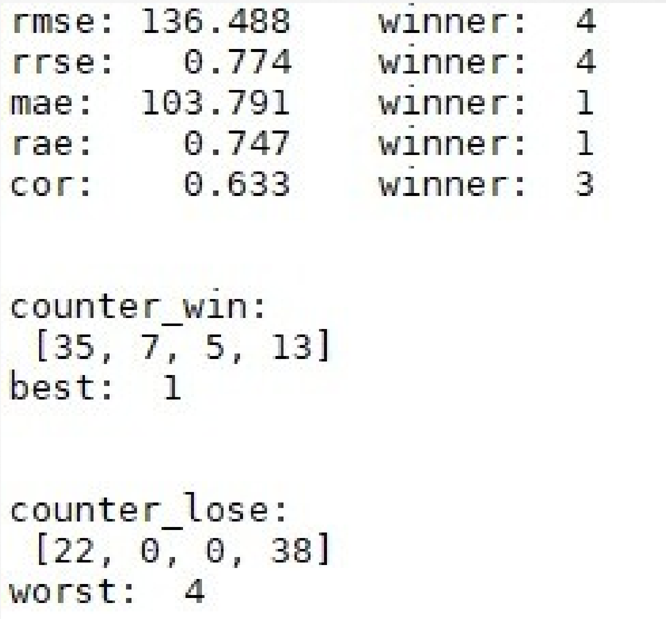
```
rmse: 142.687    Preprocessing winner:  1
rrse:   0.791    Preprocessing winner:  1
mae:  106.429    Preprocessing winner:  4
rae:    0.761    Preprocessing winner:  4
cor:    0.612    Preprocessing winner:  1


counter_win:
 [35, 6, 5, 14]
best Preproccesing:  1


counter_lose:
 [9, 4, 1, 46]
worst Preproccessing:  4
```

BikeSharing - Lasso Regression

- Generally similar to Linear Regression
- Preprocessing:
  - No obvious best,
  - but obvious worst: Feature selection
- Parameter:
  - Very unstable, no clear-cut alpha
  - Close to 1 and close to 0:
  - performance strongly dependent on train-test split
  - Middle values: more stable but worse results (average)
-

BikeSharing - Lasso Regression

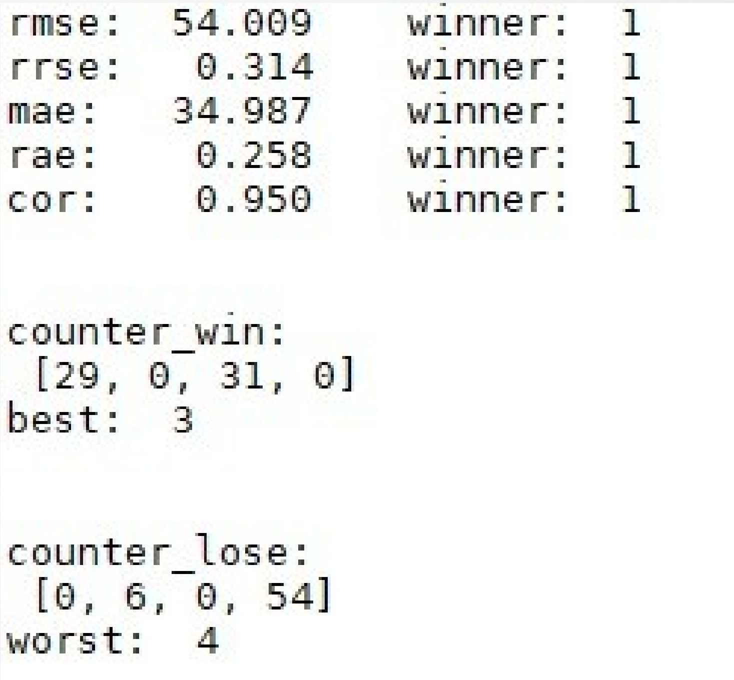
- Results: (1: alpha=1, 2: alpha=0.5, 3: alpha=0.05, 4: alpha=0.005)
  - As stated 1 and 4 often winner, often loser
  - 2,3 stable in the middle
  - Rooted mean squared error
  - slightly better than linear



```
rmse: 136.488    winner:  4
rrse:   0.774    winner:  4
mae:  103.791    winner:  1
rae:    0.747    winner:  1
cor:    0.633    winner:  3


counter_win:
 [35, 7, 5, 13]
best:  1


counter_lose:
 [22, 0, 0, 38]
worst:  4
```

BikeSharing - kNN

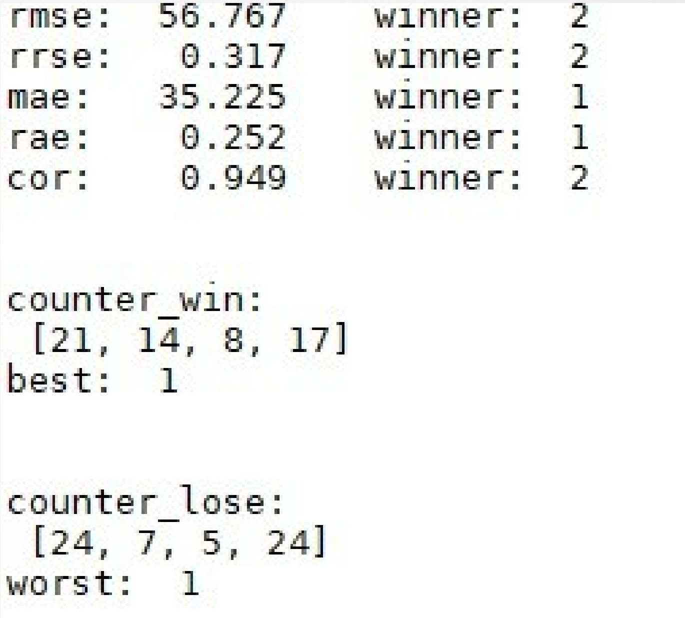
- Preprocessing
  - Less features preferred -> Feature Selection outperforms others
- Parameter weights:
  - Weigths: distance by far better
- In plot:
  1. K = 5, weights = distance
  2. K = 5, weights = uniform
  3. K = 8, weights = distance
  4. K = 8, weights = uniform
- Wins shared amongst weigths=distance



```
rmse:  54.009    winner:  1
rrse:   0.314    winner:  1
mae:   34.987    winner:  1
rae:    0.258    winner:  1
cor:    0.950    winner:  1


counter_win:
 [29, 0, 31, 0]
best:  3


counter_lose:
 [0, 6, 0, 54]
worst:  4
```

BikeSharing - kNN

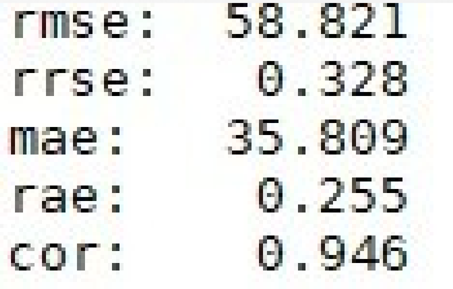
- Parameter k:
  - Easy to see: behaves badly for k<6, and k>9
  - Inbetween not too clear
- Plot: (weights=uniform, algorithm=ball_tree)
  1. K = 6
  2. K =7
  3. K = 8
  4. K = 9
- Results:
  - RMSE significantly better than for Linear/Lasso Regression



```
rmse:   56.767    winner:  2
rrse:    0.317    winner:  2
mae:    35.225    winner:  1
rae:     0.252    winner:  1
cor:     0.949    winner:  2


counter_win:
 [21, 14, 8, 17]
best:  1


counter_lose:
 [24, 7, 5, 24]
worst:  1
```

BikeSharing - Regression Tree

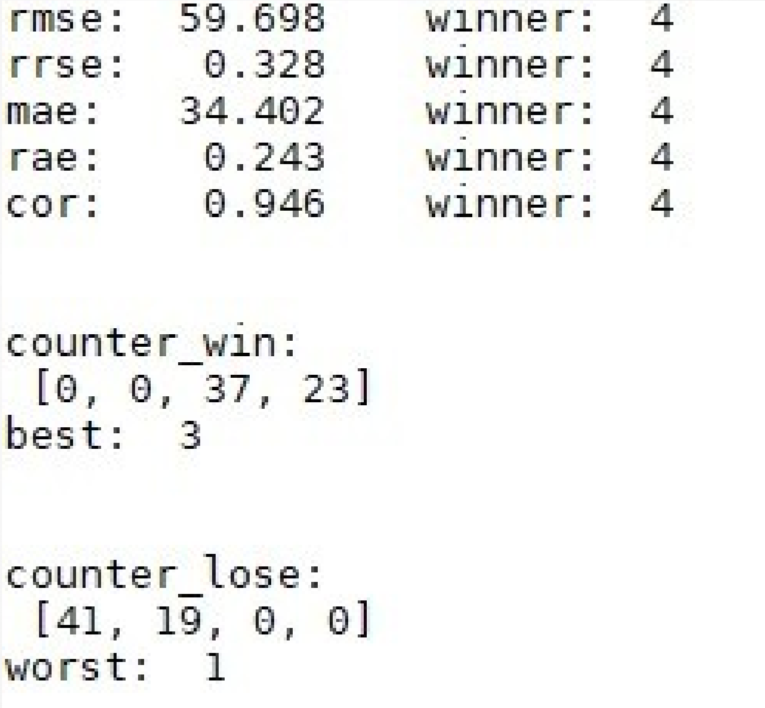
- Preprocessing:
  - Feature Selection worst
  - Others similar
- Parameters:
  - Good values for Min_samples_leaf: 2 or 3
  - Max_depth: default setting best
- Results: for raw data (option 1)
  - Similar to kNN



```
rmse:   58.821
rrse:    0.328
mae:    35.809
rae:     0.255
cor:     0.946
```

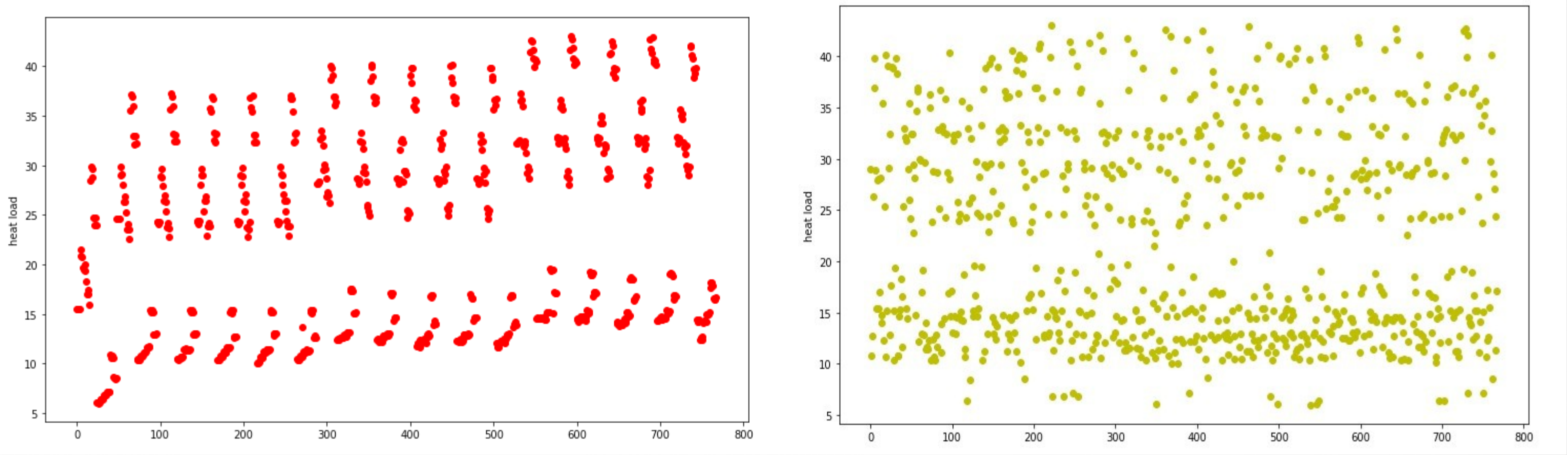
BikeSharing Method Comparison

- Plot:
  1. Linear + raw data
  2. Lasso (alpha = 0.05)+ minmax scaling
  3. kNN (8, distance) + Feature selection
  4. Tree (min_samples_leaf=2)+ raw data
- Results:
  - Knn wins slightly before Tree
  - Linear and Lasso poor performance
  - Note: winner is 4 in last iteration
- Apparently cnt not linearly dependent on features



```
rmse:   59.698     winner:   4
rrse:    0.328     winner:   4
mae:    34.402     winner:   4
rae:     0.243     winner:   4
cor:     0.946     winner:   4


counter_win:
 [0, 0, 37, 23]
best:   3


counter_lose:
 [41, 19, 0, 0]
worst:   1
```

Energy Dataset

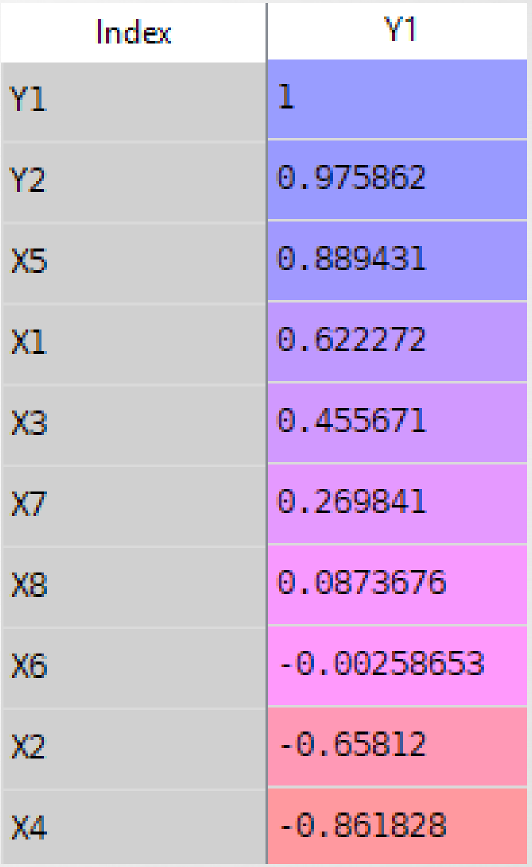


- 768 samples:
    - no missing values
    - simulated dataset
- 10 features:
    - 2 target values: Y1 and Y2
    - has only numeric values, X6 being ordinal
    - performed regression only on Y1 representing heat load (mean = 22.3, std = 10)
    - attribute X6 contains int values from 2-6,refering to position
    - dataset is ‚pregrouped' into blocks of 4, differing within a block only by X6
    - some features have a range from 0-1, others 500-800

Energy Preprocessing

- High correlation to X5 and Y2, which is not used
- very low correlation for X6 ↔ feature selection
- Normalizing <- due to different ranges

- Preprocessing options:
  1. raw data
  2. Z-score scaling
  3. minmax scaling
  4. feature selection: dropping X6 and Z-score scaling
- Observation:
  - for every method Feature Selection is best

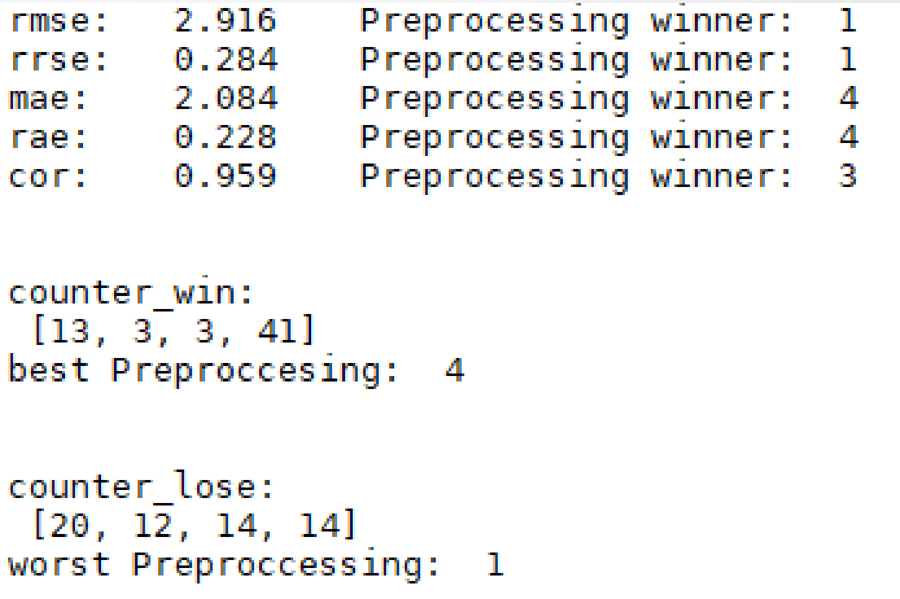


Index	Y1
Y1	1
Y2	0.975862
X5	0.889431
X1	0.622272
X3	0.455671
X7	0.269841
X8	0.0873676
X6	-0.00258653
X2	-0.65812
X4	-0.861828

Energy - Linear Regression

- Preprocessing:
  - best option is feature selection
  - worst option: raw data

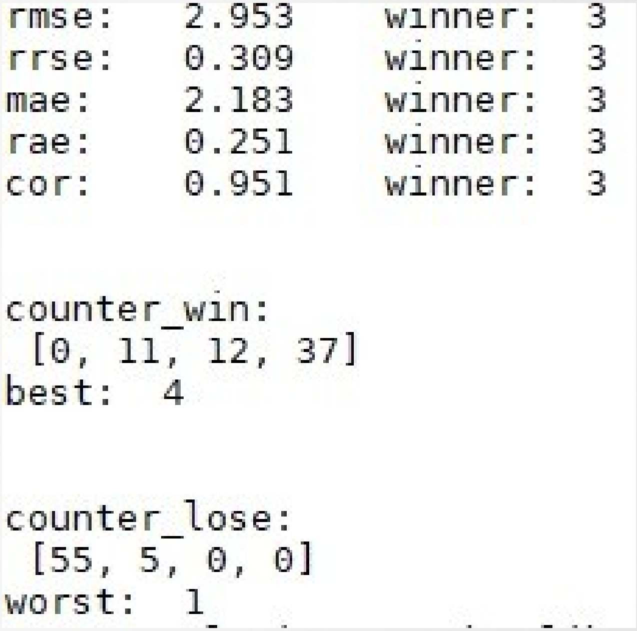
- Results:
  - rooted mean squared error = 2.92
  - rooted relative squared error = 28%
  - not good, especially compared to the
  - other methods



```
rmse:    2.916    Preprocessing winner:   1
rrse:    0.284    Preprocessing winner:   1
mae:     2.084    Preprocessing winner:   4
rae:     0.228    Preprocessing winner:   4
cor:     0.959    Preprocessing winner:   3


counter_win:
 [13, 3, 3, 41]
best Preproccesing:  4


counter_lose:
 [20, 12, 14, 14]
worst Preproccessing:  1
```

Energy - Lasso Regression

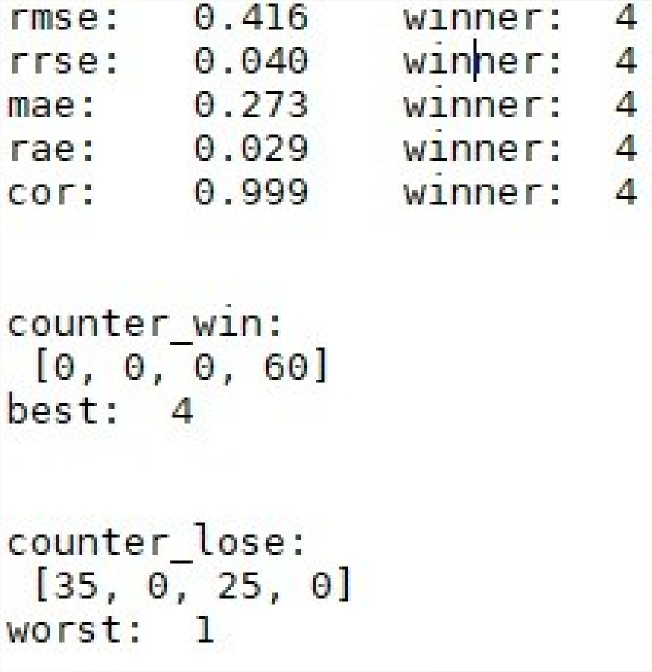
- Preprocessing
  - Best Feature Selection, Z-score scaling
  - Worst Raw
- Parameter
  - For Feature Selection: small alpha good (~0.005)
  - For Z-score Scaling: big alpha good (~0.5)
  - Together: Feature Selection + small alpha better
- Result: alpha=0.005 and variable preprocessing
  - Average: slightly better than linear, still not great



```
rmse:    2.953    winner:  3
rrse:    0.309    winner:  3
mae:     2.183    winner:  3
rae:     0.251    winner:  3
cor:     0.951    winner:  3


counter_win:
 [0, 11, 12, 37]
best:  4


counter_lose:
 [55, 5, 0, 0]
worst:  1
```

Energy - kNN

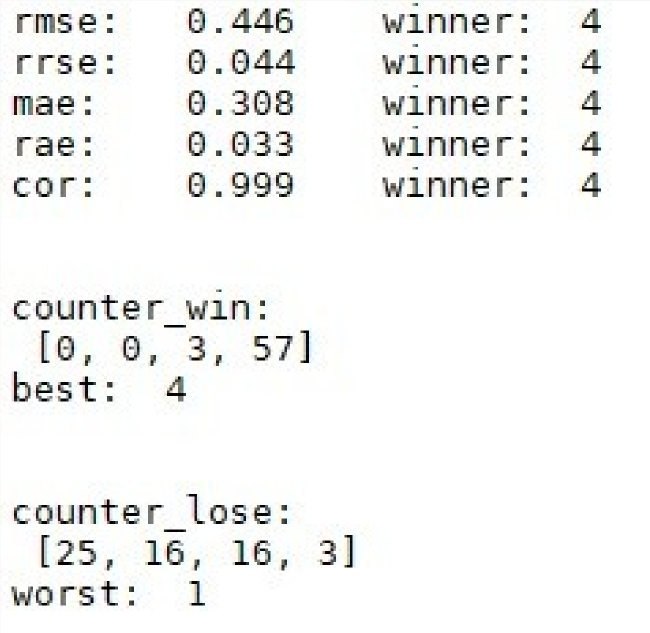
- Preprocessing
  - Feature Selection best
  - Raw and minmax perform poorly
- Parameter
  - Best value for k=3 (4 very similar, 3 intuitive best)
  - Weights= distance (fits intuition)
  - Optimal parameter values easy to find
- Results
  - RMSE drastically improved compared to Linear and Lasso
  - Correlation: 99,9% (!)



```
rmse:    0.416     winner:  4
rrse:    0.040     winner:  4
mae:     0.273     winner:  4
rae:     0.029     winner:  4
cor:     0.999     winner:  4


counter_win:
 [0, 0, 0, 60]
best:  4


counter_lose:
 [35, 0, 25, 0]
worst:  1
```

Energy - Regression Tree

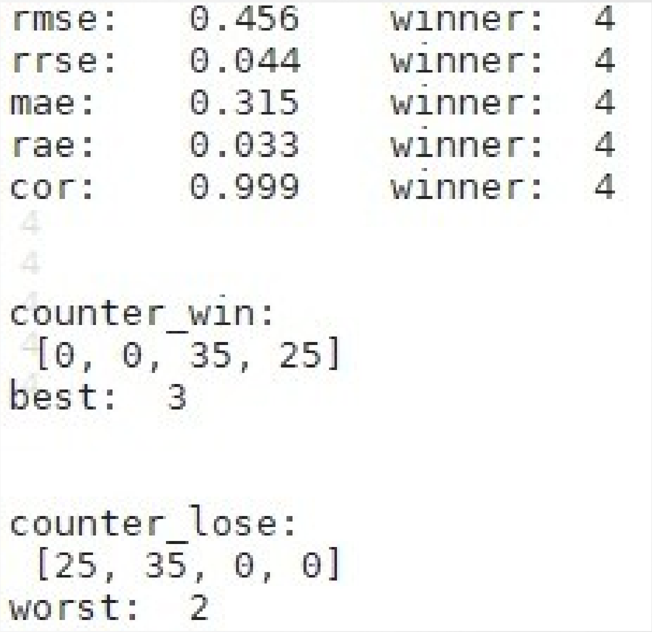
- Preprocessing
  - Even here Feature Selection outperforms others
- Parameters
  - Min_samples_leaf=2 to smoothen the model
  - Max_depth= default more stable
- Results min_samples_leaf=2, max_depth=default
  - Also very good results
  - Slightly less stable than kNN



```
rmse:    0.446    winner:  4
rrse:    0.044    winner:  4
mae:     0.308    winner:  4
rae:     0.033    winner:  4
cor:     0.999    winner:  4


counter_win:
 [0, 0, 3, 57]
best:  4


counter_lose:
 [25, 16, 16, 3]
worst:  1
```

Energy Method Comparison

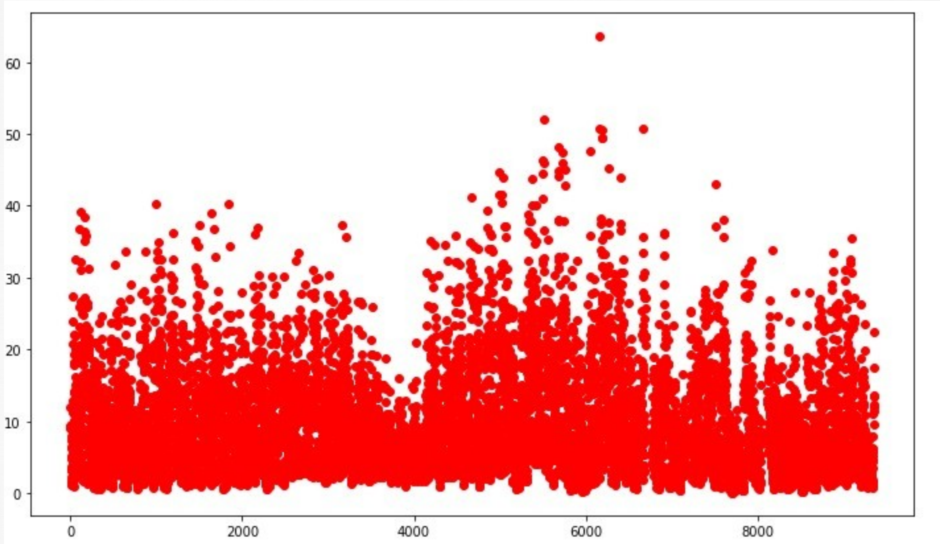
- Graphic
  1. Linear + Feature Selection
  2. Lasso (0.05) + Feature Selection
  3. kNN (k=3, weights=distance) + Feature Selection
  4. Tree (min_samples_leaf=2) + Feature Selection
- Results
  - kNN and Regression Tree share wins
  - kNN best ->fits intuition:
  - every datapoint has 3 very close neighbors
  - Linear and Lasso share losses



```
rmse:    0.456    winner:  4
rrse:    0.044    winner:  4
mae:     0.315    winner:  4
rae:     0.033    winner:  4
cor:     0.999    winner:  4
 4
 4
counter_win:
 [0, 0, 35, 25]
best:  3


counter_lose:
 [25, 35, 0, 0]
worst:  2
```

AirQuality Dataset

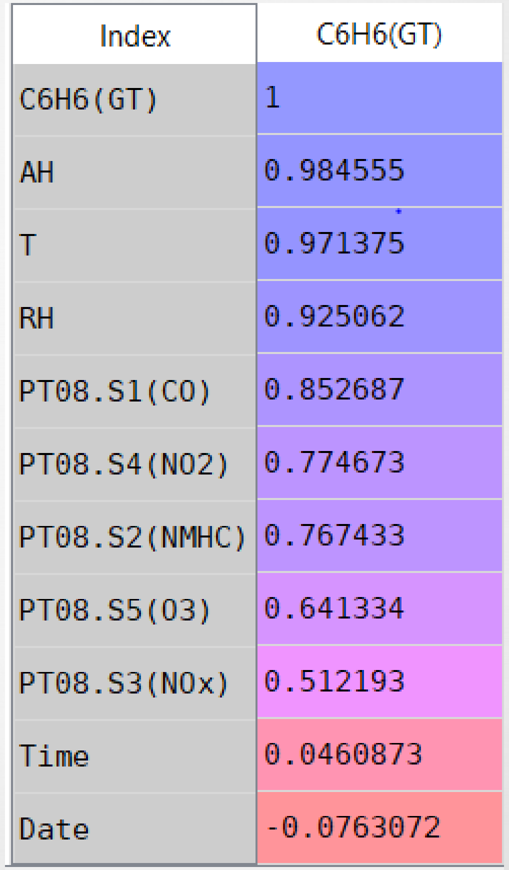
- Features (after cleaning up dataset)
    - 10 float/ordinal values
  - Converted date and time into ordinal values
  - Ranges
  - 0-1200 for sensor data, AH between 0-1.2
  - → Normalization necessary



- Target Value
  - Benzene value in italian city: 0-50 C_6H_6
  - Mean=1.9 (-200 values!) , median=7.9 , std= 41
- Missing values – Yes, set to value -200
- Size of dataset: 9357 samples

AirQuality Preprocessing

- Correlation Features <-> Target
  - Very high correlation for every chemical info feature
  - Very low for time and data
  - →idea: drop them
- Missing values
  - Left as -200
  - No significant improvement by
  - treating missing values (mean,median,...)



Index	C6H6(GT)
C6H6(GT)	1
AH	0.984555
T	0.971375
RH	0.925062
PT08.S1(CO)	0.852687
PT08.S4(NO2)	0.774673
PT08.S2(NMHC)	0.767433
PT08.S5(O3)	0.641334
PT08.S3(NOx)	0.512193
Time	0.0460873
Date	-0.0763072

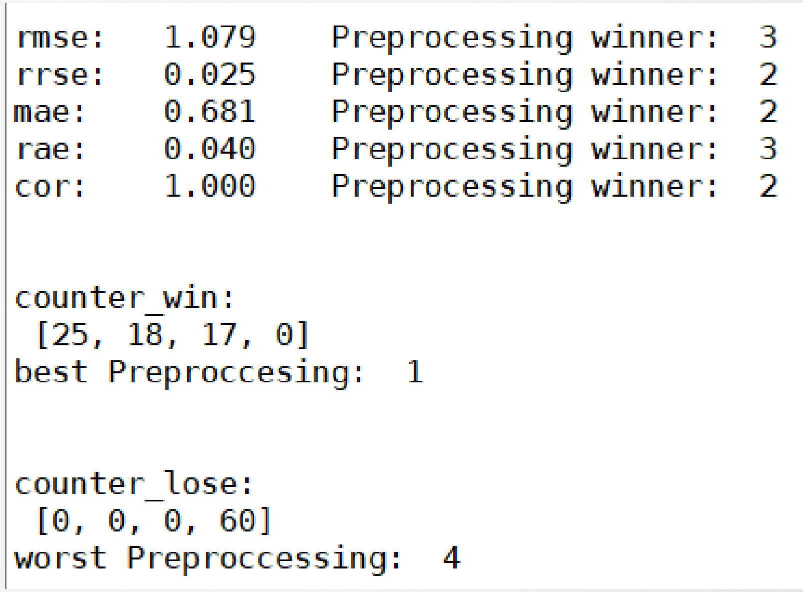
AirQuality Preprocessing

- Preprocessing options
  1. Raw data
  2. Z-score scaling
  3. Minmax scaling
  4. Feature Selection: drop time and date
  5. + z-score scaling
- Surprisingly Feature Selection often worst Preprocessing approach

AirQuality - Linear Regression

- Preprocessing
  - Raw data performs best
  - Feature selection worst by far
  - Scaling doesn‘t change much

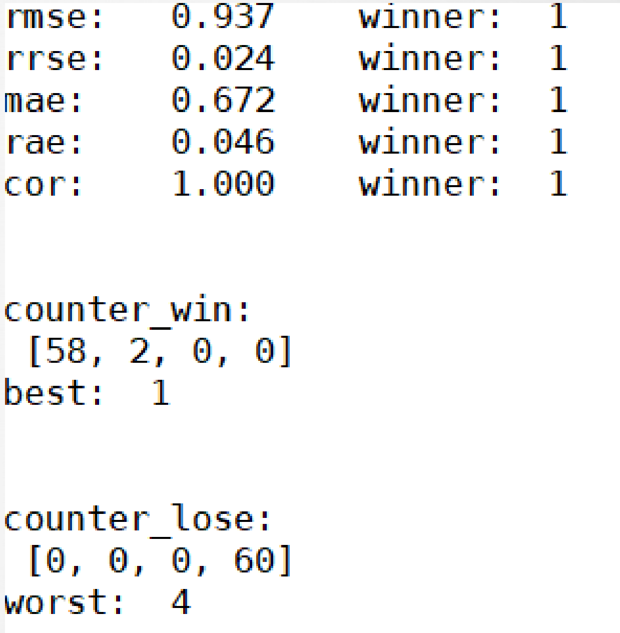
- Results
  - Actually RMSE not bad, but
  - other methods better
  - RMSE=1.1
  - Relative RMSE 2.5%



```
rmse:    1.079    Preprocessing winner:  3
rrse:    0.025    Preprocessing winner:  2
mae:     0.681    Preprocessing winner:  2
rae:     0.040    Preprocessing winner:  3
cor:     1.000    Preprocessing winner:  2


counter_win:
 [25, 18, 17, 0]
best Preproccesing:  1


counter_lose:
 [0, 0, 0, 60]
worst Preproccessing:  4
```

AirQuality - Lasso Regression

- Preprocessing
  - Very clear: best is raw data, worst is Feature Selection
  - Other two similar
- Parameter
  - The smaller the alpha the better
  - Graphics (alpha=0.005, alpha=0.05, alpha=0.25, alpha=0.5):
  - obviously smaller alpha better
- Results
  - Performs slightly better than linear Regression
  - RMSE 0.94
  - Relative error again about 2.4%



```
rmse:     0.937     winner:   1
rrse:     0.024     winner:   1
mae:      0.672     winner:   1
rae:      0.046     winner:   1
cor:      1.000     winner:   1


counter_win:
 [58, 2, 0, 0]
best:   1


counter_lose:
 [0, 0, 0, 60]
worst:   4
```

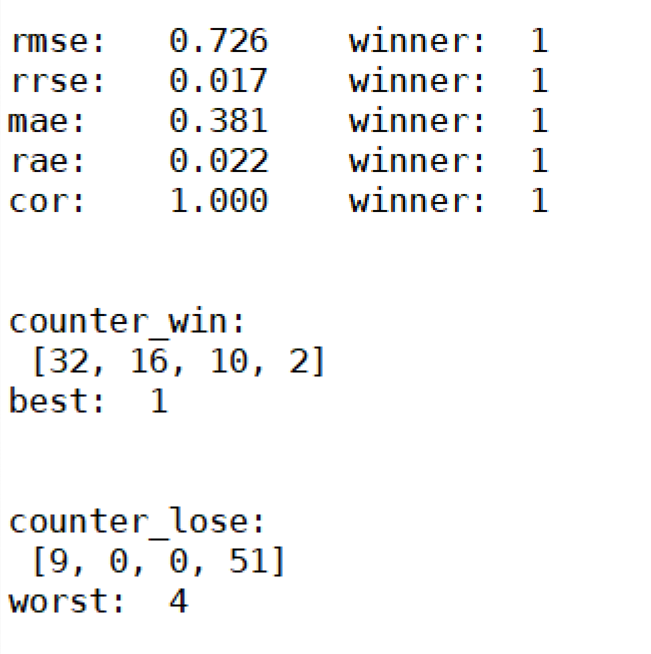
AirQuality - kNN

- Preprocessing
  - Both Raw data and Feature Selection good
  - Worst by far: Minmax -> strong outliers
  - Interestingly also worst when replacing -200 by mean
  - Reason?
    - Values vary drastically
    - shrinking down to [0,1] leads to insignificant distance
    - Too small distance, even though values far from each other

- Parameter weights Raw data
  - Weights=distance always outperformed uniform

AirQuality - kNN

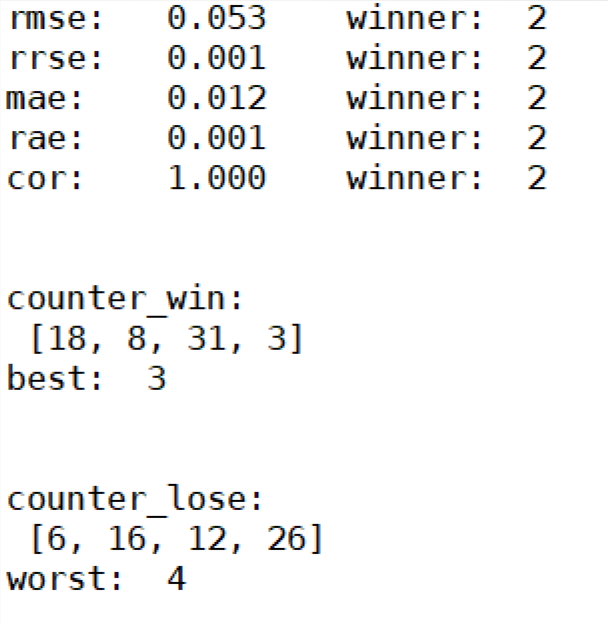
- Parameter k Raw data
  - k: values from 4-6 good,
  - 4 instable, often best, sometimes worst
- Plot Raw data, weigths=distance
  1. K=4
  2. K=5
  3. K=6
  4. K=7
- Results
  - Very satisfying results!
  - Relative error of 1.7%
  - Correlation very close to 1



```
rmse:    0.726    winner:  1
rrse:    0.017    winner:  1
mae:     0.381    winner:  1
rae:     0.022    winner:  1
cor:     1.000    winner:  1


counter_win:
 [32, 16, 10, 2]
best:  1


counter_lose:
 [9, 0, 0, 51]
worst:  4
```

AirQuality - RegressionTree

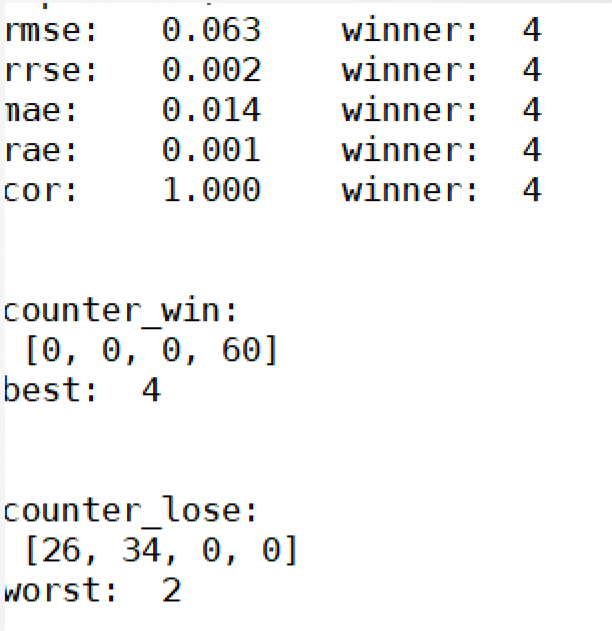
- Preprocessing
  - As for kNN: minmax performs very poorly
  - Best one is Feature Selection
- Parameter
  - Min_samples_leaf = 2 yields good results,
  - considering 4: samples may be grouped,
  - that are not similar
  - Max_depth a little more stable using 12
- Results 1: (2,12), 2: (4,12), 3: (2,13), 4: (4,13)
  - Both 1 and 3 good, 3 more unstable
  - Excellent results! Relative error of 0.1%



```
rmse:    0.053    winner:  2
rrse:    0.001    winner:  2
mae:     0.012    winner:  2
rae:     0.001    winner:  2
cor:     1.000    winner:  2


counter_win:
 [18, 8, 31, 3]
best:  3


counter_lose:
 [6, 16, 12, 26]
worst:  4
```

AirQuality Method Comparison

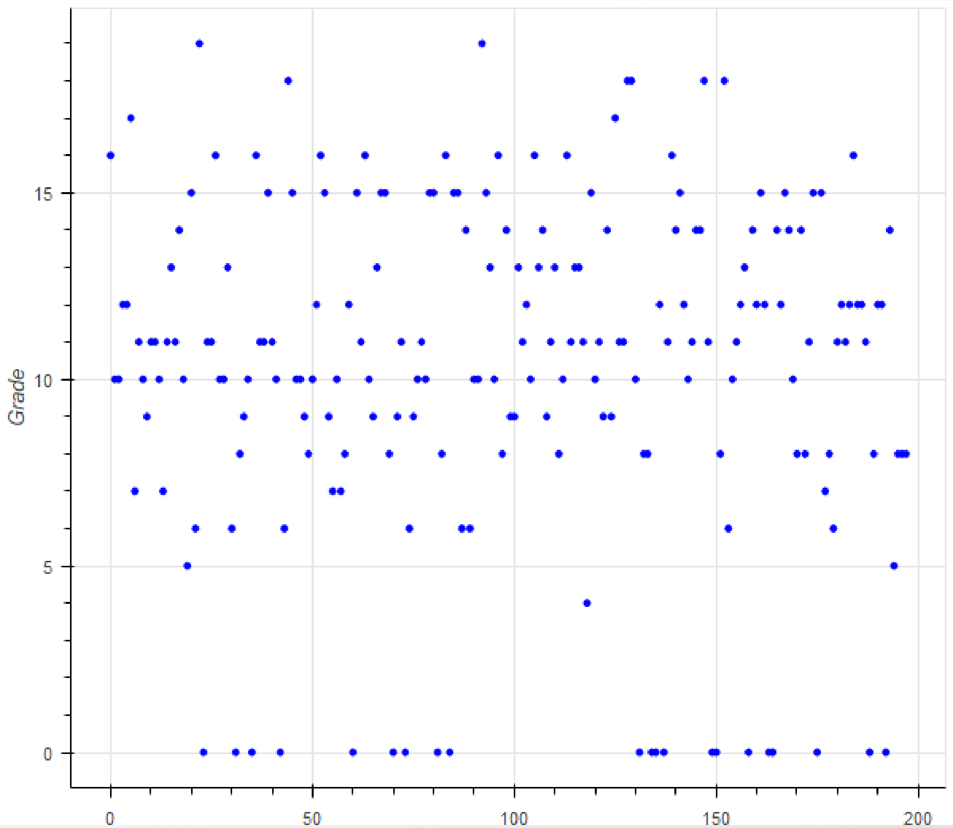
- Graphic:
  1. Linear + Raw data
  2. Lasso (alpha=0.005) + Raw data
  3. kNN (k=5, distance) + Raw data
  4. Tree (min_samples_leaf=2, max_depth=12)
  5. + Feature Selection
- Results
  - Very clear results: Tree by far the best
  - kNN second
  - Linear and Lasso last
  - Altogether very nice error rates



```
rmse:    0.063    winner:  4
rrse:    0.002    winner:  4
mae:     0.014    winner:  4
rae:     0.001    winner:  4
cor:     1.000    winner:  4


counter_win:
 [0, 0, 0, 60]
best:  4


counter_lose:
 [26, 34, 0, 0]
worst:  2
```

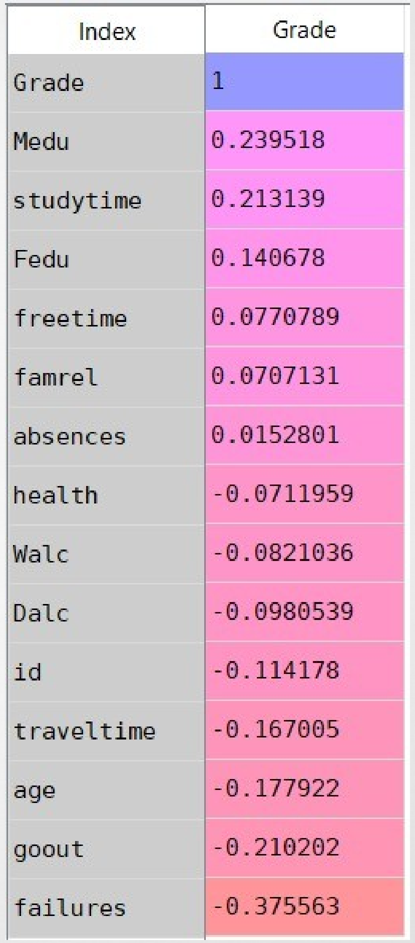
Students Dataset

- 198 samples:
  - no missing values
- 30 features:
  - Several Boolean & ordinal features, no numeric
  - Lots of booleans -> bad performance of scaling
  - Many features with low correlation to target value
- Target value: school grades
  - Value 0-20
  - Mean=10,3 , median=11, std=4,6



Students Preprocessing

- Low correlation for most features
  - Extremely low for: absences, health, Walc, Dalc
  - No correlation for categorical data
    - Used our ordinal encoding
- Preprocessing options:
  1. OneHotEncoding, others left raw
  2. OneHotEncoding + z-score scaling
  3. oneHotEncoding + minmax scaling
  4. Handwritten ordinal encoding (intuitive order)
     1. z-score scaling
     2. Feature Selection (dropping of every feature with correlation <0.01)
- 4th Approach for every method the best

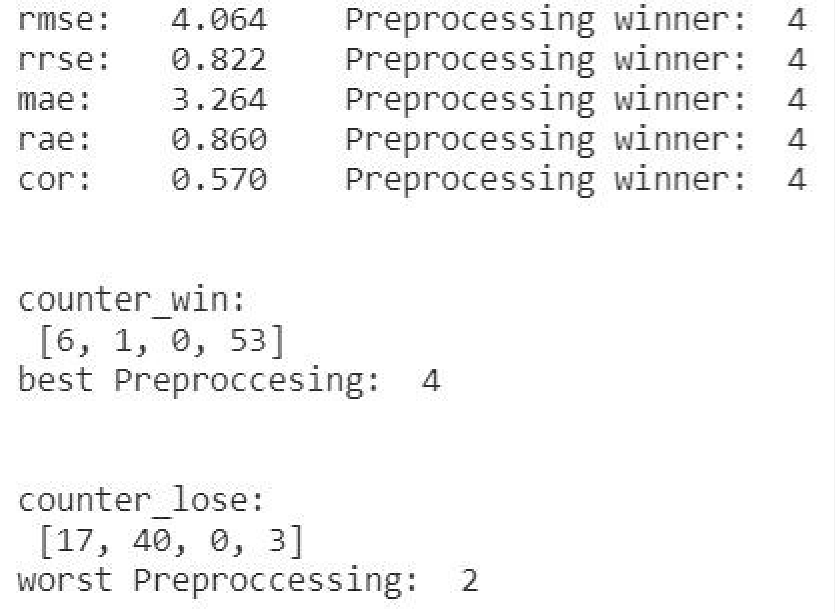


Index	Grade
Grade	1
Medu	0.239518
studytime	0.213139
Fedu	0.140678
freetime	0.0770789
famrel	0.0707131
absences	0.0152801
health	-0.0711959
Walc	-0.0821036
Dalc	-0.0980539
id	-0.114178
traveltime	-0.167005
age	-0.177922
goout	-0.210202
failures	-0.375563

Students - Linear Regression

- Preprocessing
  - Handmade ordinal encoding is best by far (human intuition is taken into account)
  - Raw data and MinMax similarly bad

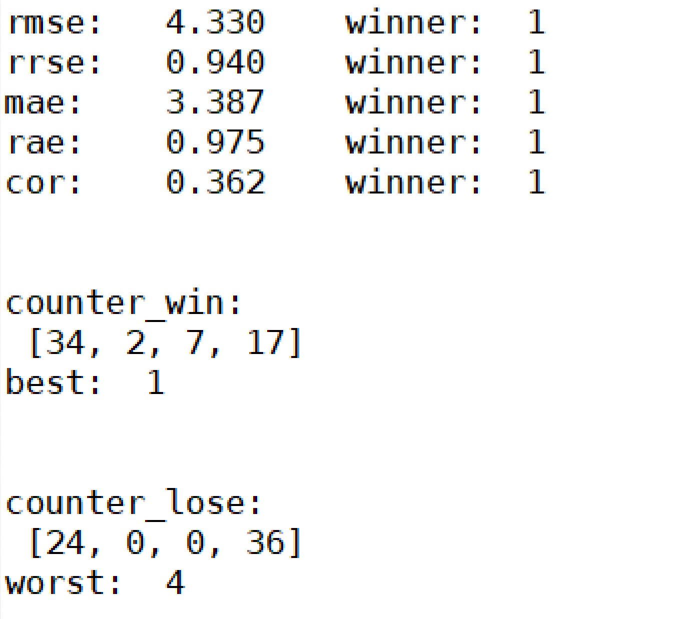
- Results
  - Rooted mean squared error = 4.1
  - Rooted relative squared error 82%
  - Linear is quite good compared to other methods



```
rmse:   4.064    Preprocessing winner:  4
rrse:   0.822    Preprocessing winner:  4
mae:    3.264    Preprocessing winner:  4
rae:    0.860    Preprocessing winner:  4
cor:    0.570    Preprocessing winner:  4


counter_win:
 [6, 1, 0, 53]
best Preproccesing:  4


counter_lose:
 [17, 40, 0, 3]
worst Preproccessing:  2
```

Students - Lasso Regression

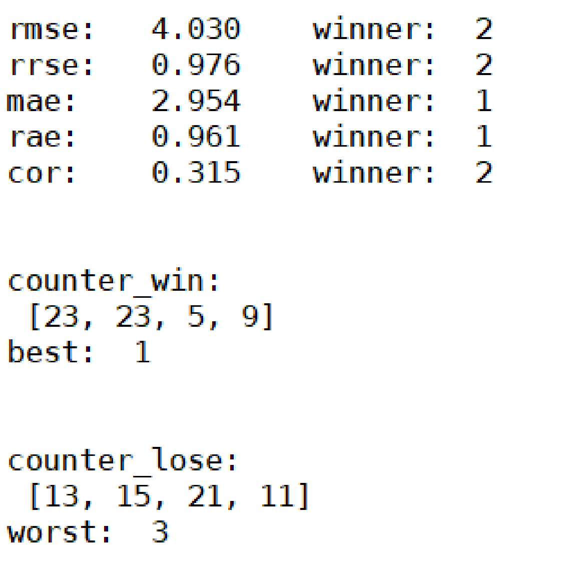
- Preprocessing
  - Similar to Linear Regression
- Parameters
  - Alpha between 0.15 and 0.3 best
  - In plot: 1: alpha=0.15, 2: alpha=0.2, 3:alpha=0.25, 4: alpha=0.3
  - 0.15 and 0.3 quite instable though
  - Fix alpha=0.25 for further uses
- Results
  - Rooted mean square error = 4.3
  - Rooted relative square error = 0.94
  - Lasso also quite good / same level as linear



```
rmse:    4.330    winner:  1
rrse:    0.940    winner:  1
mae:     3.387    winner:  1
rae:     0.975    winner:  1
cor:     0.362    winner:  1


counter_win:
 [34, 2, 7, 17]
best:  1


counter_lose:
 [24, 0, 0, 36]
worst:  4
```

Students - kNN

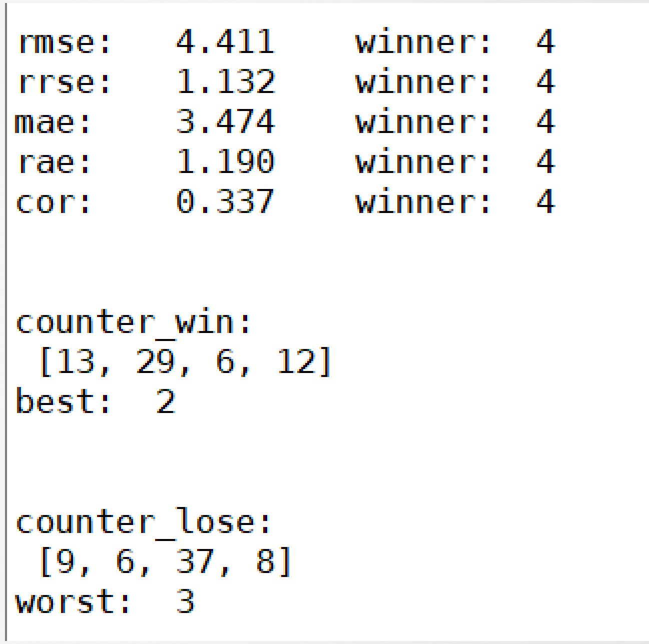
- Preprocessing: see Linear and Lasso
- Parameter weights
  - Quite clear: uniform outperforms distance
  - Fits intuition: many dimensions, distance may not be significant
  - In graphic: 1 and 2 uniform (k=20,30), 3 and 4 distance (k=20,30)
- Parameter k
  - No definite result
  - There is no best k
  - Every value from 24-40 okay
- Results
  - In this graphic a little better than Linear and Lasso
  - In average slightly worse



```
rmse:    4.030    winner:  2
rrse:    0.976    winner:  2
mae:     2.954    winner:  1
rae:     0.961    winner:  1
cor:     0.315    winner:  2


counter_win:
 [23, 23, 5, 9]
best:  1


counter_lose:
 [13, 15, 21, 11]
worst:  3
```

Students - Regression Tree

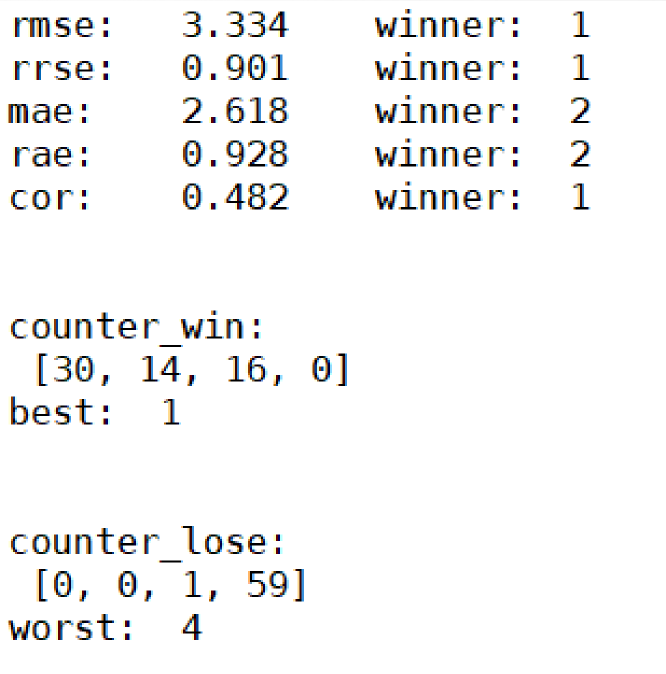
- Preprocessing
  - OrdinalEncoding wins
  - Raw data the worst
  - MinMax and z-score scaling similar
- Parameters
  - Surprisinlgy, decreasing max_depth to 6
  - increases performance
    - Even though a small dataset
  - Also min_samples_leaf = 3 is quite definite the best
- Results 1: (2,6), 2: (3,6), 3: (2,default), 4: (3,default)
  - RMSE of 4.4 is worse than all the other methods
  - Small datasets not suitable for RegressionTrees



```
rmse:    4.411    winner:  4
rrse:    1.132    winner:  4
mae:     3.474    winner:  4
rae:     1.190    winner:  4
cor:     0.337    winner:  4


counter_win:
 [13, 29, 6, 12]
best:  2


counter_lose:
 [9, 6, 37, 8]
worst:  3
```

Students Methode Comparison

- Plot
  1. Linear + Ordinal
  2. Lasso (0.25)+ Ordinal
  3. Knn (28, uniform)+ Ordinal
  4. Tree (3,6) + Ordinal
- Results
  - Best performance: Linear Regression!
  - Worst by far: Regression Tree
  - Unfortunately still a relative error of 90%
  - Conclusio: kNN and Regression Tree cannot handle very small datasets well



```
rmse:     3.334    winner:  1
rrse:     0.901    winner:  1
mae:      2.618    winner:  2
rae:      0.928    winner:  2
cor:      0.482    winner:  1


counter_win:
 [30, 14, 16, 0]
best:  1


counter_lose:
 [0, 0, 1, 59]
worst:  4
```

Comparison of Results

- Predictions for Energy and Air good
- For Bike and Students rather poor
- Knn prefers less features, cannot handle many booleans well
- Tree prefers more features, stable wrt data type and range
    - Tree uses random choice -> same split, different results
- Linear and Lasso only good for students (size of dataset?, linear dep?)
- Preprocessing almost always usefull
- kNN and Regression tree can handle clustered data well
- kNN and Regression tree cannot handle small datasets well