Henrik Sainio

Aalto University

Effects of dimensionality reduction with PCA and feature selection in regression and classification performance using weather data

Abstract

This report studies how dimensionality reduction with Principal Component Analysis and feature selection can be used with machine learning models and how it affects the model performance. The study uses weather data that consists of 3140 samples and 16 features collected in a weather station in Helsinki from September 2006 to May 2019.

The task is to use linear regression and quadratic regression to predict relative humidity based on observations, find the significant predicting features and examine the change in metrics and model performance after dimensionality reduction with Principal Component Analysis (PCA) and feature selection. In addition, K-nearest Neighbors and Logistic Regression are used to classify whether the weather conditions are “dry” or “not dry” and the accuracy of the methods are compared after reducing dimensionality with PCA and feature selection.

We find that in regression the features with most prediction power are essential in our models’ performance. Since principal components represent the variation in the data, there is a possibility that the significant features are pushed back in the components with least variation, and that dimensionality reduction with PCA does not undoubtedly improve the model performance.

In classification problem with logistic regression and k-nearest neighbor, the performance was improved after dimensionality reduction. Best result was achieved with just 5 principal components applied to 12 of the original features. The individual features in the data did not seem to hold as much weight to accuracy score as the variance given by principal components. Discarding the obsolete features with feature selection is also necessary, since the models might suffer from the curse of dimensionality and unnecessary features decrease model interpretability as well as training speed.

Going forward, the models could be further improved by introducing more preprocessing, cross-validating and adding more data samples.

1. Introduction

This report is motivated by the final project assignment for Aalto University course “Data Science”. The course offers an introduction to the field of data science and machine learning with tools to extract and learn information from data with statistical methods.

Completing this project is a great way to put the theoretical knowledge of the methods to use with a realistic dataset and a relevant problem. I am hoping to grasp a strong understanding of the methods I am using in this project and to continue learning more complex machine learning models, data visualization and data processing. Furthermore, this assignment will be conducted with such motivation that it can be further extended to be used as a portfolio showcase.

Predicting weather from collected data is an excellent opportunity to study machine learning. Since meteorological prediction models are highly demanding and out of the scope of this report, the problems and model complexity are simplified to fit for a machine learning training material.

2. Data analysis

2.1 Description of data

The weather data has been cleaned with no null-values, duplicate samples or irrelevant observations. The data is split into training data with 3140 samples, 16 input features and two target labels for both regression and classification tasks, and testing data with 1346 samples.

All input features are numerical and the mean (e.g. **"T_mu"**) and the variance (**"T_var"**) values for each day are included in the dataset.

Numerical feature definitions:

"T" is the air temperature, in degrees Celsius, 2 meters above the earths surface.

"Po" is the atmospheric pressure at weather station level, in millimetres of mercury.

"P" is the atmospheric pressure reduced to mean sea level, in millimetres of mercury.

"Ff" is the mean wind speed at a height of 10-12 meters above the earths surface, in meters per second.

"Tn" is the minimum air temperature, in degrees Celsius, over the past day.

"Tx" is the maximum air temperature, in degrees Celsius, over the past day.

"Vv" is the horizontal visibility, in kilometers.

"Td" is the dewpoint temperature at a height of 2 meters above the earths surface, in degrees Celsius.

"U" is the relative humidity, in percentage, 2 meters above the earth's surface.

Target label definitions:

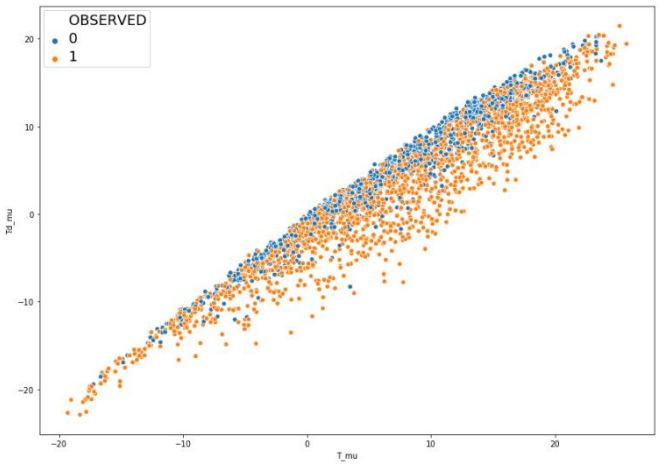
"OBSERVED" is a categorical variable, where 0 (not dry) indicates that the amount of precipitation was more than 0.3 millimetres, and 1 (dry) indicates that there was little or no precipitation.

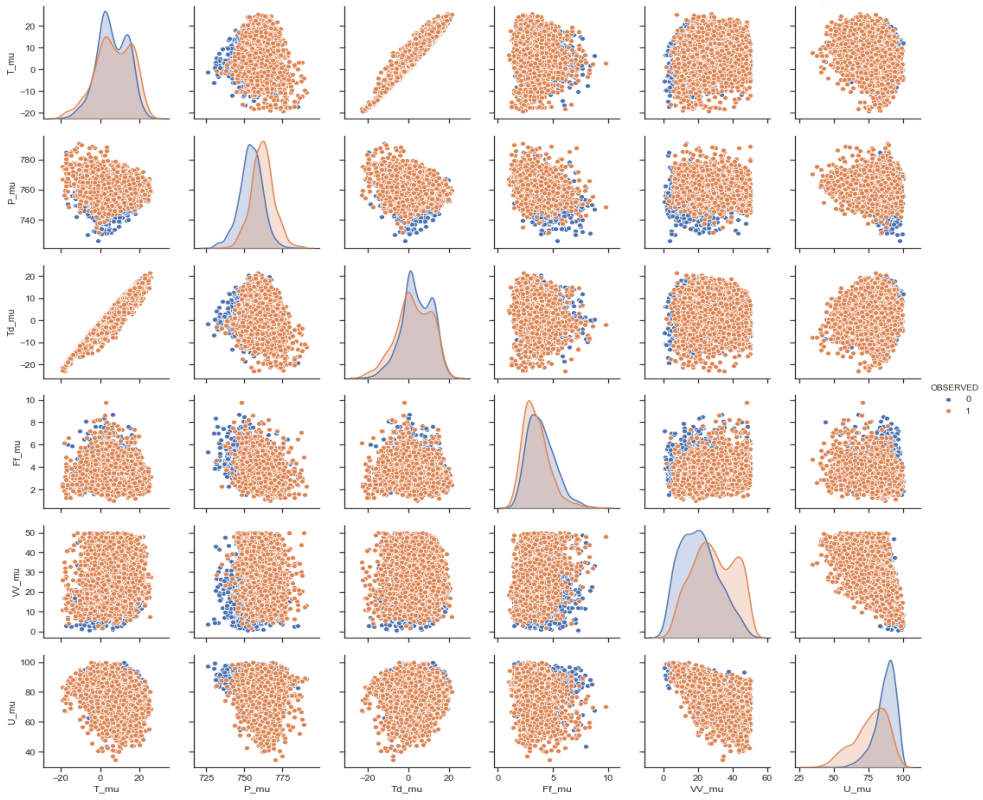
"U_mu" is the relative humidity.

2.2 Data exploration

Plotting some of the most important features is an important part of understanding the data and it usually gives us some information about the most significant features and the features with most correlation. Pair plots show pairwise relationships between six selected features and the univariate distributions on the diagonal line.

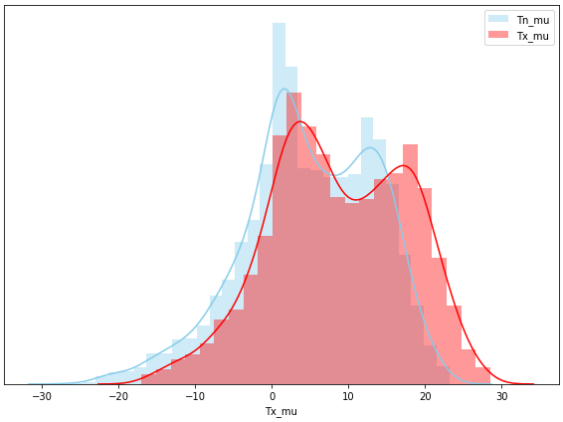
Dew point temperature and air temperature have very strong linear correlation and in regression task we can assume these features have significant effect on the predictions. We also notice that data points classified as 0, follow a tight pattern that might be very useful for fitting the linear regression line.





The relative humidity “U_mu” is correlated strongly with the horizontal visibility “VV”. When relative humidity is high, we can expect the visibility to decrease. Similarly, lower atmospheric pressure is correlated with more observed rainy weather. Computationally these might still not be the most important features to note.

Parallel distribution plot of minimum and maximum air temperature. As expected, the histograms are similar in shape and shifted a few degrees as the temperature changes during the day.



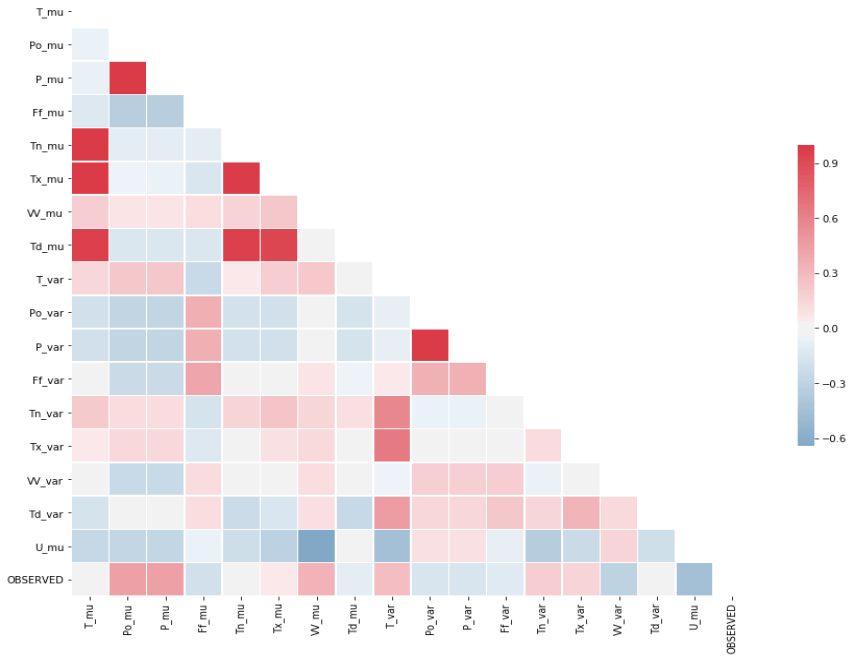
The correlation matrix represents the statistical relationship between two random variables. From standardized data correlation matrix is calculated as follows,

$$\boldsymbol{S} = \frac{\boldsymbol{1}}{\boldsymbol{n-1}} \cdot \boldsymbol{D^T D}$$

where **D** is the data matrix and **n** is the number of samples.

Correlation values vary from [-1,1]. Features are positively correlated when they either increase or decrease together and negatively correlated when one value decreases as the other one increases.

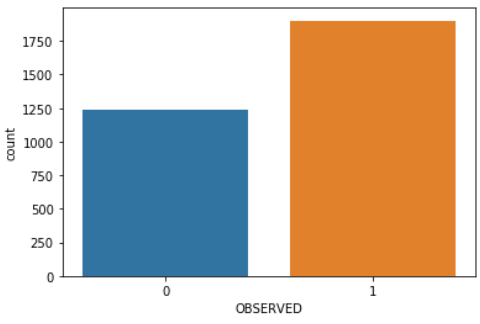
In correlation matrix we see strong positive correlation in min and max air temperatures and mean temperature as well as atmospheric pressures. As above mentioned, the visibility and relative humidity are negatively correlated.



In classification imbalanced class distribution can affect the accuracy of the model for the following few key reasons.

Some learners factor in prior classes (the marginal probabilities of the classes) that in imbalanced classes might be heavily biased in favor of the majority class. This leads to more misclassified minory class predictions.

The learner might also not be able to draw the boundaries for the minority class as well as for the majority class in case of too few minority class samples.



3. Methods

3.1 Preprocessing

Original data did not include any missing data or remarkable outliers. Data was already split into training and validation data.

For optimal performance of principal component analysis, the data was standardized resulting in rescaled features so that

$\mu=0$ and $\sigma=1$, where μ is the mean and σ is the standard deviation.

Standardization is then calculated with

$$z = \frac{x-\mu}{\sigma}$$

Generally, in classifier models adding dimensionality will improve the performance until optimal number of dimensions is found. However, further increasing dimensionality will have the opposite effect if we don't increase the number of data samples since the data becomes increasingly more sparse and less meaningful. Training the model with too high dimensionality will result in overfitting. The model starts to learn the data too well, including exceptions that are specific to the training data. This leads to overblown training data accuracy and severely decreased validation data accuracy.

Principal Component Analysis, PCA, will solve this issue by creating new components holding as much information from the original data as possible in lower dimensionality. New components are linear combinations from original features and represent the most variation in the data. Components are selected so that the first component shows the most variation and the further components in descending order show the rest of the variation.

Calculating the components follow this simplified path:

1. Standardize the given data to 0-mean and to standard deviation of 1

We standardize and/or normalize the data because often features are measured in different scales and it can have a substantial effect on calculated variances and Euclidian distances in clustering tasks.

2. Compute the covariance matrix of the data

If the data is standardized, the covariance matrix is the same as the correlation matrix

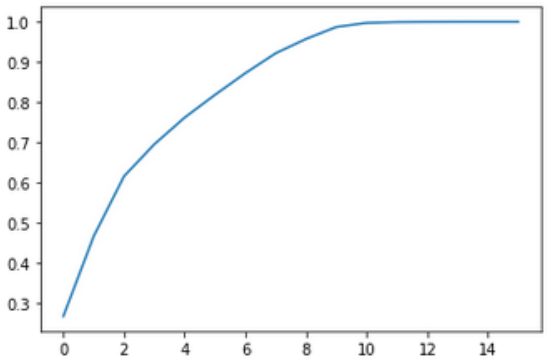
$$\boldsymbol{S = D^T D}$$

3. Calculate the eigenvectors of the covariance matrix and the corresponding eigenvalues. Sort the eigenvectors and -values in a descending order and create a sorted eigenvector matrix **W**

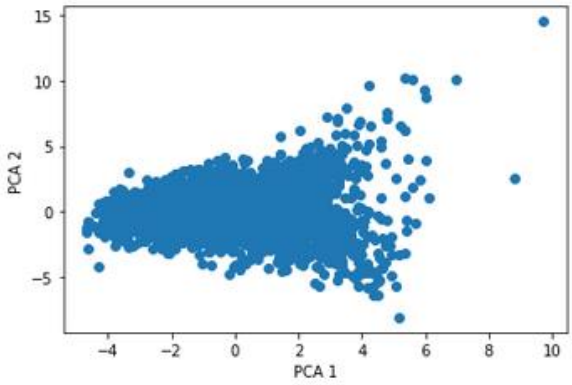
Largest eigenvectors of the covariance matrix of the data show the orthogonal direction of largest variance. Dimension reduction can from here on be done by selecting a **k**-amount of principal components and dropping the eigenvectors that represent the least of variation.

Question of how many components we should choose, can be answered with *explained variance* calculated with eigenvalues. Explained variance tells us how much each component represent the original variance. Cumulative explained variance can be used to determine how many components we need to achieve, for example with 8 components we cover 90% of the original variance.

The projection of components shows the first component variance on x-axis and second component variance on y-axis.



Cumulative explained variance



Projection of two first components

4. Calculate the component matrix $\mathbf{Y}=\mathbf{X}\times\mathbf{W}$,

where **Y** is the new data with our transformed samples, **X** is the original data with **n** samples and **W** is the **n × k** eigenvector matrix.

3.2 Regression and classification models

3.2.1 Linear regression

Multiple linear regression models the relationship between input variables and a target variable by fitting a linear equation to the data.

In linear models we assume that target variable can be calculated from linear combinations of input variables.

In a simple linear regression task with one input, the form of the model is

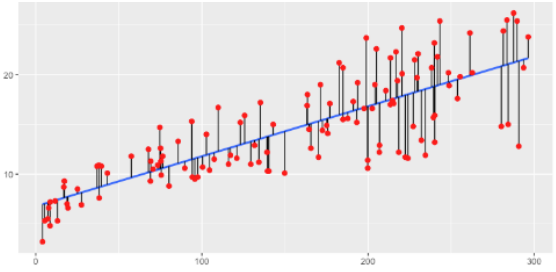
$$\boldsymbol{y = mX + c + \epsilon}$$

,where **m** and **c** are the regression coefficients and some error term $\boldsymbol{\epsilon}$.

Multidimensional form of linear regression with multiple input features has the form

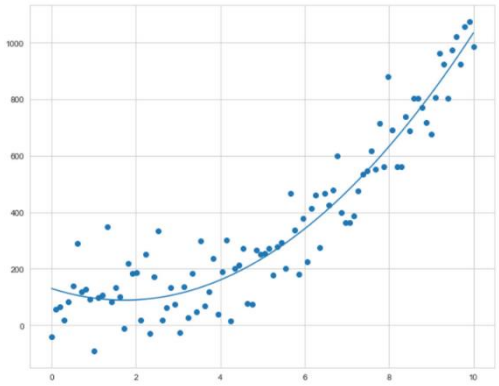
$$\boldsymbol{y_i = m_0 + m_1 X_{i1} + \ m_2 X_{i2} \ + \cdots + m_p X_{ip} + \epsilon}$$

Least squared is a procedure used to find the best fit of the regression line to the data with a loss function. It calculates vertical distance from the real data points to the regression line, squares it and minimizes the sum of all these squared residuals, thus finding the values for **m** and **c**.



3.2.2 Polynomial regression

Polynomial regression follows the same idea than linear regression (linear with respect to the coefficients of the model) but the fitted regression line is a curve.



For quadratic regression we can model the target value **y** as a second-degree polynomial,

$$y_i = m_0 + m_1 X_i + m_2 X_i^2 + \epsilon$$

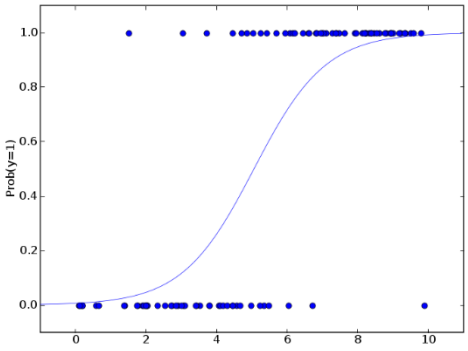
3.2.3 Logistic Regression

Logistic regression is a classification model that maps regression output to probabilities with sigmoid function

$$y' = \frac{1}{1 + e^{-(z)}}$$

Where **y'** is the regression output and

$$z_i = m_0 + m_1 X_{i1} + m_2 X_{i2} + \cdots + m_p X_{ip} + \epsilon$$



Logistic regression output is then compared against a probability threshold and the system classifies the sample to either class accordingly. If the value is above the threshold, it is classified as the positive class.

3.2.4 K-nearest Neighbors (KNN)

K-nearest Neighbors classifier is easy to implement and understood but gets slower as the amount of data samples grow.

KNN finds the distance (often Euclidean distance) between a target sample and all training points and classifies it on the most frequent class based on **k**-amount of nearest training points.

The optimal **k**-amount is usually data dependent. Incrementally increasing the **k**-value can be a good way to determine the value with highest accuracy. The higher the **k**-value the smoother the decision boundary gets and the model starts to under-fit the data. If you include neighbors from far away, you introduce some irrelevant information. On the contrary, small **k**-values have more variance and the predictions can be affected by noise and the outliers of data.

4. Results

4.1 Regression

A popular metric to analyze a regression model is the *Mean Squared Error*, MSE. It calculates the average squared difference between the estimated values and the actual value.

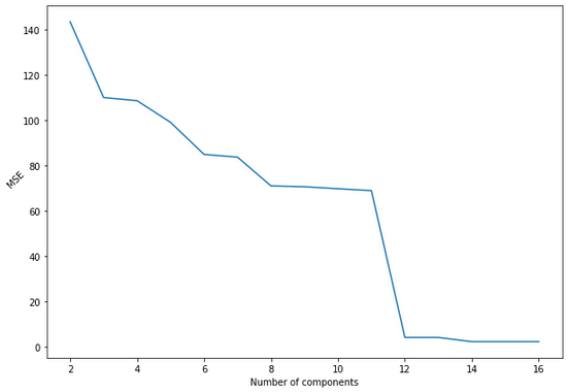
R-squared is a statistical measure of how well the regression line fits the data. Maximum value is 1, which represents perfect fit to data.

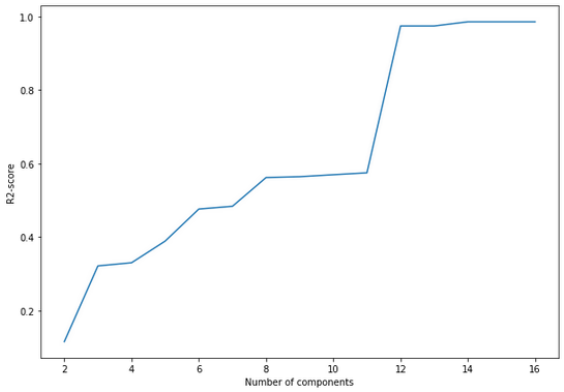
After the preprocessing steps, fitting the original data without PCA with Linear Regression we achieve these results:

Mean squared error: *2.4296* **R2_score:** *0.9850*

For this dataset, where the humidity we are predicting have values from 35 to 100, the basic linear model already gives us a decent value of average squared error. The R-squared value is very near its maximum value of 1, which tells us the model is fitting the data almost perfectly.

We discovered earlier that 8 PCA components was enough to represent >90% of the variance in the dataset. These graphs show the relationship between number of principal components used in linear regression model in a standardized dataset and the respective MSE and R-squared values. PCA sorts the new features from the one that shows most variance to the one showing least variance. However, these new features might have a great variance without being significant in predicting outputs and some features pushed back in the components might be very important to the predictions. This explains the sudden performance increase when we include 12th PCA component to our model.





With 12 components we achieve

Mean squared error: *4.2660* **R2_score:** *0.9736*

Dropping obsolete features is another way to reduce dimensionality, but we must have features that don't affect the predictions heavily. In weather dataset I was able to drop down from 16 variables to just 6 variables and still hold MSE of *3.0728* and R-squared value of *0.9810* without applying PCA. This implicates that there are indeed just a few features that hold most of the prediction power of the model.

4.2 Classification

Binary classification performance can be evaluated with model accuracy and confusion matrices.

Accuracy is the percentage of correct predictions and confusion matrix shows the different combinations of predicted and true values with true and false negatives and positives.

Logistic regression implemented to original standardized and normalized data with all features yields results

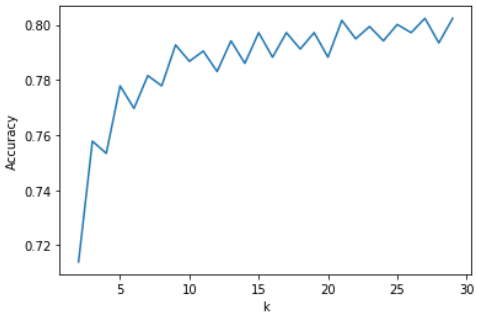
Accuracy: *0.76* Confusion matrix: $\begin{bmatrix} 278 & 150 \\ 164 & 754 \end{bmatrix}$

Applying both feature selection and PCA to using 10 of the original features and just 8 principal components we achieve a slightly improved performance with faster computation speed.

Accuracy: *0.79* Confusion matrix: $\begin{bmatrix} 308 & 120 \\ 158 & 760 \end{bmatrix}$

where we notice that most of the improvement happens to true and false positives, and in our case more correct predictions of rain and less predictions of no rain when it actually is raining. This is because irrelevant data can have negative effect on classification model performance.

Our dataset size allows us to use *elbow method* to determine the optimal k-parameter for KNN. For larger datasets and computationally limited situtations elbowing is not recommended. We might not always want to find the best accuracy by sacrificing costly computation speed.



When **k**=21 and we apply KNN to the original standardized and normalized data,

Accuracy: *0.80* Confusion matrix: $\begin{bmatrix} 305 & 123 \\ 144 & 774 \end{bmatrix}$

KNN already performs very well in general and slightly better than logistic regression. The model usually suffers from the curse of dimensionality and discarding obsolete features might improve the score even more.

With PCA applied to 12 of the original features, using 5 principal components and **k**=21 we achieve

Accuracy: *0.81* Confusion matrix: $\begin{bmatrix} 312 & 116 \\ 141 & 777 \end{bmatrix}$

Normalization, or min-maxing the data, improved the accuracy. Classification models that use distances between data points benefit from normalizing the data to the same scale.

5. Conclusions

Predicting relative humidity, air temperature and dew point temperature seemed to hold much of the prediction power. Dropping the features **together** increased the average square error and decreased R-squared value. This could be due to their natural mathematical correlation:

"When the dew point equals the air temperature, the air is saturated, and the relative humidity is %100" ([1], Steve Ackerman, University of Wisconsin-Madison)

"For every 1 °C difference in the dew point and dry bulb temperatures, the relative humidity decreases by 5%, starting with RH = 100% when the dew point equals the dry bulb temperature." [2]

Reducing dimensionality did not improve the model, but in case of larger dataset and a strict computational speed and memory limit, the model still performed well enough to be deemed useful.

We do achieve almost perfect regression line and a very small values of MSE with linear regression. Thus a quadratic model can be considered to overfit the model and possibly weaken the ability to predict future data instances.

In the classification problem the results of logistic regression are already satisfying, but discarding the obsolete features and applying dimensionality reduction with PCA improve the metrics of the model. Since we don't have a continuous value to predict, as we have in a linear regression task, the individual feature dependencies in the data seemed not to have as much significance as the variance given by principal components.

6. References

[1] http://cimss.ssec.wisc.edu/wxwise/rh/page2.html

[2] https://www.jstor.org/stable/26221248?seq=1

https://www.researchgate.net/publication/2364670_The_Effect_of_Class_Distribution_on_Classifier_Learning_An_Empirical_Study

https://sebastianraschka.com/Articles/2014_about_feature_scaling.html

https://www.datacamp.com/community/tutorials/k-nearest-neighbor-classification-scikit-learn

https://sebastianraschka.com/Articles/2014_intro_supervised_learning.html

http://mathworld.wolfram.com/topics/Regression.html