Snowfall Depth Predictor

Objective:

To predict the depth of the snowfall received by the area on the given day

Approach:

1. Data Overview:

Firstly, Pandas dataframes are created by reading the given csv files. Then, a brief overview of the training data is obtained with the help of data statistics like count, mean, std, min, max, 50 percentile, etc. and histogram representations of the various features. It is observed that there are no missing values.

1. Encoding Date-time features:

It is observed that the ‘DATE’ and ‘MST’ features are of object data type. So, it is essential to extract features like ‘Month’, ‘Day’ and ‘Hour’ from them. Keeping in mind the cyclic nature of months, days and hours, these features are accordingly encoded using sine and cosine transformations. The following formulae are used:

data[x+‘sin’] = sin(2\*pi\*data[x]/max\_val)

data[x+‘cos’] = cos(2\*pi\*data[x]/max\_val)

1. Data Cleaning:

Purposeless features like Serial No., ‘DATE’, ‘MST’, ‘Month’, ‘Day’ and ‘Hour’ are removed from both train and test data.

During data overviewing, negative values of ‘Snow Depth’ were observed, which are clearly human error or error in the measuring instrument. So, it is essential to replace these values with NULL. Then the SimpleImputer class of sklearn.impute is used to impute those NULL values with median of the remaining data.

1. Train Test Splitting:

Train test splitting is an integral part of the machine learning model. It enables us to estimate the performance of our ML algorithm. The StratifiedShuffleSplit class of sklearn.model\_selection splits the entire dataset into train and test dataset by preserving the composition of each sample in the specified feature (‘Hour\_sin’ in this case).

1. Finding correlations:

Correlation is the measure of the extent to which two variables are associated. Here, we measure the correlation between ‘Snow Depth [cm]’ with all the features.

For plotting the graph for correlation, scatter plots are used. Scatterplots are plotted between ‘Snow Depth [cm]’ and the features which seem to show relatively strong correlation.

Another way to plot the correlation matrix is by using a heatmap where different shades of colours are used to visualize the strength of correlation.

1. Trying out attribute combinations for Strong Correlation:

Since all the existing features do not show strong correlation, we introduce a new feature ‘Temperature/Pressure [deg C/mBar]’ which is calculated as follows:

Temperature/Pressure [deg C/mBar] = Tower Dry Bulb Temperature [deg C]/Station Pressure [mBar]

It is observed that this new feature has a negative correlation which is stronger compared to existing features.

1. Creating a Pipeline:

Pipeline is a model through which a series of transformations like imputing and feature scaling can be performed.

Imputing is the process by which missing values of NaN type can be filled with a measure of central tendency like mean, median or mode.

Feature Scaling is the process to normalize the range of feature data. StandardScaler class of sklearn.preprocessing transforms the features data so that it ranges from 0 to 1 by using the following transformation:

data[x] = (data[x] - mean)/std

1. Selecting a desired model:

There are a number of regression techniques such as Linear Regression, Polynomial Regression, Decision Tree Regression, Random Forest Regression, Support Vector Regression, Logistic Regression, etc. Some of these techniques are tried out one by one.

1. Evaluating the model:

For all the different regression techniques, the ML model is evaluated by calculating the RMSE (root mean squared error) on the train data itself.

1. Cross Validation:

In order to check if overfitting has not taken place, Cross-Validation is performed. It is a method that uses different portions of the data to train and test a model on different iterations and returns the Cross-Validation scores (which is a measure of relative error) for different iterations.

1. Testing the model on test data:

The model, hence chosen on the basis of lesser CV score, is tested on the test data. Firstly, the ‘Snow Depth [cm]’ column is copied to a separate NumPy array for reference, then it is dropped from the test dataset. The dataset is made to undergo the same transformations using the pipeline as created above.

1. Saving the model:

Saving our model enables us to load it later in order to make predictions. A model can be saved in two ways: using pickle and using joblib. Joblib is module of SciPy which saves and loads Python files consisting of NumPy data structures.

1. Submitting Predictions:

The model is loaded from the saved joblib file and used to make the final predictions. The predictions are converted from a NumPy array to a Pandas dataframe which is saved as a CSV file.