Do we need more bikes?
Project in Statistical Machine Learning

Anonymous Author(s)
Affiliation
Address
email

Abstract

In this project we develop, and study different statistical machine learning models for predicting whether the number of available bikes at a given hour should be increased, a project by the District Department of Transportation in Washington D.C. The training data set consists of 1600 instances of hourly bike rentals, and a test set of 400 instances. The models for prediction we have used are: *Logistic regression, Discriminant methods: LDA, QDA, k- Nearest Neighbour*, and *Tree Based Methods*. We have found that THE MODEL gives best prediction, with accuracy ??????

1 Plan

1.1 From Intro

(i) Explotre and preprocess data

(ii) try some or all classification methods, which are these?

- Logistic Regression
- Discriminant analysis: LDA, QDA
- K-nearest neighbor
- Tree-based methods: classification trees, random forests, bagging
- Boositing

(iii) Which of these are to be "put in producion"?

1.2 From Data analysis task

- Can any trend be seen comparing different hours, weeks, months?
- Is there any diffrence between weekdays and holidays?
- Is there any trend depending on the weather?

1.3 From Implementation of methods

Each group member should implement one family each, who did what shall be clear!

DNNs are encouraged to be implemented, do this if there is time. (DNN is not a thing a group member can claim as their family.)

Implement a naive version, let's do: *Always low_bike_demand*

1.3.1 What to do with each method

1. Implement the method (each person individually)
2. Tune hyper-parameters, discuss how this is done (each person individually)
3. Evaluate with for example cross-validation. Don't use E_{k-fold} (what is that?) (need to do together)
4. (optional) Think about input features, are all relevant? (together)

Before training, unify pre-processing FOR ALL METHODS and choose ONE OR MULTIPLE metrics to evaluate the model. (is it neccesary to have the same for all?, is it beneficial?) Examples:

- accuracy
- f1-score
- recall
- precision

Use same test-train split for ALL MODELS

2 Theoretical background

2.1 Mathematical Overview of the Models

2.1.1 Logistic Regression

The backbone of logistic regression is linear regression, i.e. finding the least-squares solution to an equation system

$$X\theta = b \tag{1}$$

given by the normal equations

$$X^T X\theta = X^T b \tag{2}$$

where X is the training data matrix, θ is the coefficient vector and b is the training output. The parameter vector is then used in the sigmoid function:

$$\sigma(z) = \frac{e^z}{1+e^z} : \mathbb{R} \to [0,1], \tag{3}$$

$$z = x^T\theta, \tag{4}$$

where x is the testing input. This gives a statistical interpretation of the input vector. In the case of a binary True/False classification, the value of the sigmoid function then determines the class.

2.2 Input Data Modification

By plotting the data and analyzing the .csv file, some observations were made. The different inputs were then changed accordingly:

- *Kept as-is*: `weekday`, `windspeed`, `visibility`, `temp`
- *Modified*:
  - `month` - split into two inputs, one cosine and one sine part. This make the new inputs linear and can follow the fluctuations of the year. The original input was discarded.
  - `hour_of_day` - split into three boolean variables: `demand_day`, `demand_evening`, and `demand_night`, reflecting if the time was between 08-14, 15-19 or 20-07 respectively. This was done because plotting the data showed three different plateaues of demand for the different time intervals. The original input was discarded.
  - `snowdepth`, `precip` were transformed into booleans, reflecting if it was raining or if there was snow on the ground or not. This was done as there was no times where demand was high when it was raining or when there was snow on the ground.
- *Removed*: `cloudcover`, `day_of_week`, `snow`, `dew`, `holiday`, `summertime`, `humidity`. These were removed due to being redundant (e.g. `summertime`), not showing a clear trend (e.g. `cloudcover`) or both (e.g. `day_of_week`).