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

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

given by the normal equations

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

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.

2.1.2 Random forest

The random forest method is a based upon decision trees, i.e. dividing the data point into binary groups based on Gini-impurity, entropy or classification error, Gini being the most common. These divisions are then used to create a binary tree shown in figure **??**Tree) and where thee leaf-nodes are used to classify the target variables bases on the input. As of itself the dicition tree tends to have unsatisfying results which leads to methodes like random forest that boost its accuracy.

2.1.3 Non-parametric method: k–Nearest Neighbour

k– *Nearest Neighbour*(k–NN) is a distance based method that takes a k amount of points from the training data set, called *neighbours*, computes the distance between them, then assumes that the predicted value $\hat{y}(x_*)$ follows the trend of the k– nearest neighbours. Since k–NN uses the training data explicitly it is also called a *nonparametric* method.

The k–NN method can be divided into several subcategories, inter alias *classification* k–NN method, *regression* k–NN method. In this project, we are using the classification method, since we are trying to predict in which of the two classes low, or high demand, the given, and predicted data points belong.

The classification k–NN algorithm evaluates $\hat{y}(x_*)$ by computing the most frequently occurring class among the k nearest neighbours. Here, we try to identify whether a data point belong to the high demand-class. Denote $c =$ high demand class. For simplicity, assume Euclidean ambiance. Then

$$\hat{y}(x_*) = \arg\max_c \sum_{n\in\mathbb{N}} \chi_{(y_i=c)},$$

where y_i is the class of the nearest neighbour, χ is the characteristic function

$$\chi_{(y_i=c)} = \begin{cases} 1 & \text{if } y_n = c, \\ 0 & \text{otherwise.} \end{cases}$$

It is very common to use a weighted sum to predict the next value, i.e.

$$\hat{y}(x_*) = arg\max_c \sum_{n\in\mathbb{N}} \frac{\chi_{(y_n=c)}}{d(x,x_n)},$$

where d is the standard Euclidean metric, computing the distance between an input x, and a neighbour x_n.

2.1.4 Discriminant analysis: LDA and QDA

Linear Discriminant Analysis is a generative model, which means it is a model that's creating and using a probaility distribution $P(\mathbf{x}, y)$ to create an estimation for the probability $P(y = m|\mathbf{x})$ using bayes theorem.
Bayes theorem is:

$$p(y|\mathbf{x}) = \frac{p(y, \mathbf{x})}{p(\mathbf{x})} = \frac{p(y)p(\mathbf{x}|y)}{\int_y p(y, \mathbf{x})}$$

For the discrete version it is obtained:

$$p(y = m|\mathbf{x}) = \frac{p(y = m)p(\mathbf{x}|y = m)}{\sum_{m=1}^{M} p(y = m)p(\mathbf{x}|y = m)}$$

For this form of the equation to be useful, it is neccesary to obtain an accurate estimation of $p(y = m)$ and $p(\mathbf{x}|y = m)$ for all classes m.
In LDA, $p(y = m)$ is estimated by counting the percentage of data points (in the training data) being in each of the classes and using that percentage as the probability of a data point being in that class.
In mathematical terms:

$$p(y = m) = \frac{1}{n}\sum_{i=1}^{n} \mathbb{I}\{y_i = m\} = \frac{n_m}{n}$$

To estimete the probability distribution $p(\mathbf{x}|y = m)$, a multi-dimensional gaussian distribution is used:

$$\mathcal{N}(\mathbf{x}|\mu, \mathbf{\Sigma}) = \frac{1}{(2\pi)^{d/2}|\mathbf{\Sigma}|^{1/2}} exp\left(-\frac{1}{2}(\mathbf{x} - \mu)^T \mathbf{\Sigma}^{-1}(\mathbf{x} - \mu)\right)$$

Where $\mathbf{x}$ is the d-dimentional data point, μ is the (d-dimentional) mean of the random variable. $\mathbf{\Sigma}$ is the symetric, positive definite covariance matrix defined by:

$$\mathbf{\Sigma} = \frac{1}{n - M}\sum_{m=1}^{M}\sum_{i:y_i=m}(\mathbf{x}_i - \mu_m)(\mathbf{x}_i - \mu_m)^T$$

Using these estimations results in an expression for the quantity $p(y = m|\mathbf{x})\forall m$. LDA then uses maximum likelyhood to categorize an input $\mathbf{x}$ into a class m.

Quadratic discriminant analysis (QDA) is heavily based of LDA with the sole difference being how the covariance matrix $\mathbf{\Sigma}$ is created. In LDA, the covariance matrix is assumed to be the same for data in each and every class. In QDA however, the covariance matrix is calculated for each class as follows:

$$\mathbf{\Sigma}_m = \frac{1}{n_m - 1}\sum_{i:y_i=m}(\mathbf{x}_i - \mu_m)(\mathbf{x}_i - \mu_m)^T$$

2.2 Input Data Modification

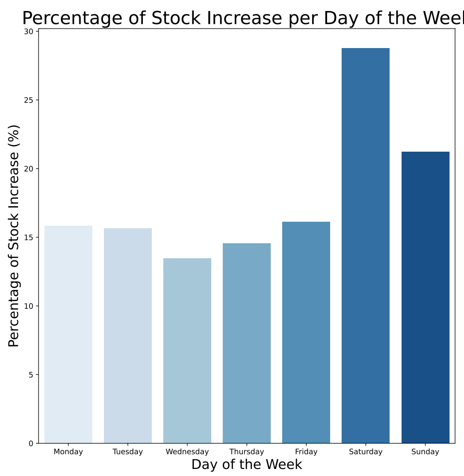
By plotting the data and analyzing the .csv file, some observations were made. The different inputs were then changed accordingly:

- *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`. These were removed due to being redundant (e.g. `summertime`), not showing a clear trend (e.g. `cloudcover`), giving a worse score when used, or all three (e.g. `day_of_week`).

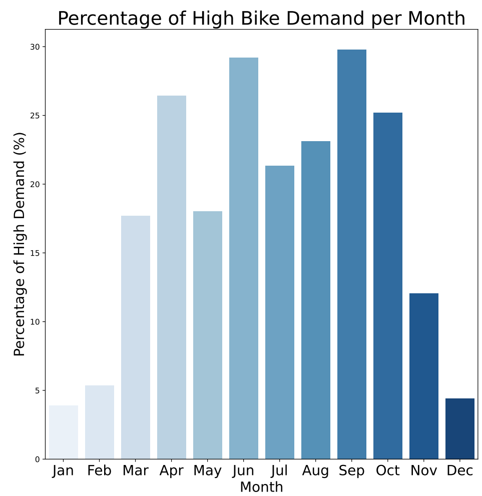
3 Data Analysis

In the data, there are some numerical and categorical features:

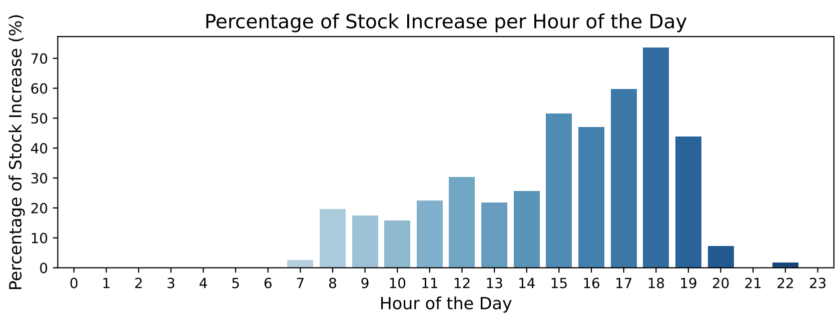
- *Numerical*: `temp`, `dew`, `humidity`, `precip`, `snow`, `snowdepth`, `windspeed`, `cloudcover` and `visibility`.
- *Categorical*: `hour_of_day`, `day_of_week`, `month`, `holiday`, `weekday`, `summertime`, and `increase_stock`



(a) Demand per day of week.



(b) Demand per month.



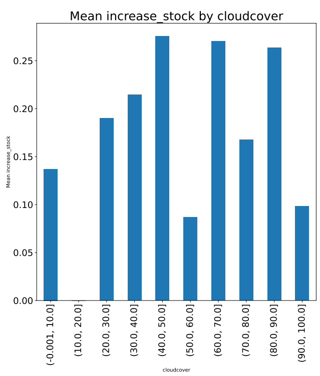
(c) Demand per hour of day.

Figure 1: Bike demand vs. day of week and month.

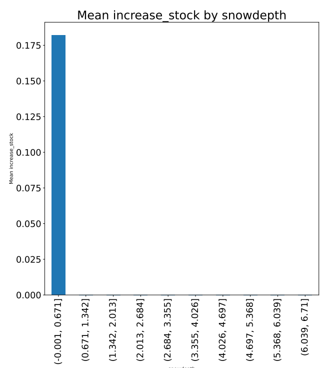
There are some trends seen in the data when it comes to time and weather. From figure 1, one can see a periodic relationship for the months, where there is a higher demand during the warmer months, loosely following a trigonometric curve. Over the week, the demand is rather stable, with a peak on the weekend, especially saturdays.

Looking at the weather (figure 2); if there is rain or if there is snow on the ground, there is close to always low demand. Cloudcover did not make a big impact, which is also intuitive, as a cloudy day does not make biking more difficult. Dew point also does not have a clear trend, while humidity however has a clear trend downwards as the humidity increases. Temperature had a more clear impact, where more people wanted to bike the warmer it got.

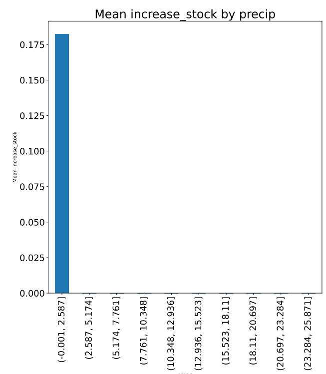
The overall trend is that about one eigth of observations correspond to a high bike demand. During the night, or in bad weather, the demand is (intuitively) low. But during rush hour (figure 1c), the demand is very high, and should probably be increased in order to minimize excessive CO_2 emissions.



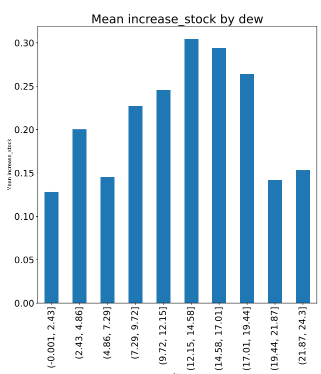
(a) Demand per cloudcover (percentage).



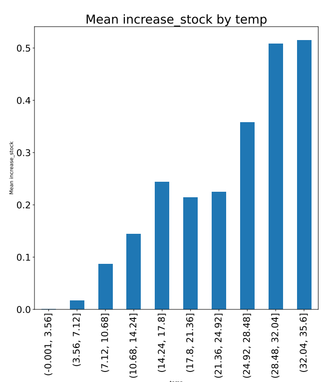
(b) Demand per day of week.



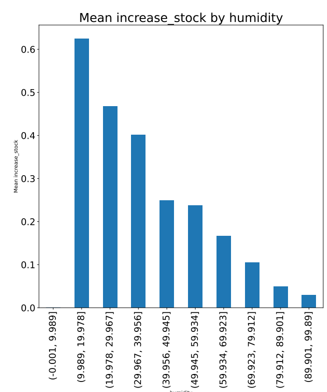
(c) Demand per day of week.



(d) Demand per dew point (°C).



(e) Demand per temperature (°).



(f) Demand per humidity level (percentage).

Figure 2: Bike demand vs. various weather parameters.