Supervised ML (regression) - Bike sharing demand prediction

Introduction

Currently, Rental bikes are introduced in many urban cities for the enhancement of mobility comfort. It is important to make the rental bike available and accessible to the public at the right time as it lessens the waiting time. Eventually, providing the city with a stable supply of rental bikes becomes a major concern. The crucial part is the prediction of the bike count required at each hour for the stable supply of rental bikes.

Problem statement

We are tasked with predicting the number of bikes rented each hour so as to make an approximate estimation of the number of bikes to be made available to the public given a particular hour of the day.

Overview of data

The dataset contains weather information (Temperature, Humidity, Windspeed, Visibility, Dewpoint, Solar radiation, Snowfall, Rainfall), the number of bikes rented per hour and date information.

Attribute Information:

- Date: year-month-day
- Rented Bike count - Count of bikes rented at each hour
- Hour - Hour of the day
- Temperature-Temperature in Celsius
- Humidity - %

- Windspeed - m/s
- Visibility - 10m
- Dew point temperature - Celsius
- Solar radiation - MJ/m2
- Rainfall - mm
- Snowfall - cm
- Seasons - Winter, Spring, Summer, Autumn
- Holiday - Holiday/No holiday
- Functional Day - NoFunc(Non Functional Hours), Fun(Functional hours)

Steps involved

I. Performing EDA (exploratory data analysis)

A. Exploring the head and tail of the data to get insights on the given data.

B. Looking for null values and removing them if it affects the performance of the model.

C. Converting the data into appropriate data types to create a regression model

D. Creating data frames that help in drawing insights from the dataset.

E. Creating more columns in our dataset which would be helpful for creating the model.

F. Encoding the string type data to better fit our regression model.

G. Calculating inter-quartile range and filtering our data.

H. Extracting correlation heatmap and calculating VIF to remove correlated and multi-collinear variables

II. Drawing conclusions from the data

Plotting a necessary graph that provides relevant information on our data like:

A. Most bikes have been rented in the summer season.

B. Least bike rent count is in the winter season.

C. autumn and spring seasons have almost equal amounts of bike rent count.

D. Most of the bikes have been rented in the year 2018.

E. Most of the bikes have been rented on working days.

F. Very few bikes have been rented in December which is the winter season.

G. Most bikes have been rented in December in the year 2017 as we don't have data before that.

H. People tend to rent bikes when there is no or less rainfall.

I. People tend to rent bikes when there is no or less snowfall.

J. People tend to rent bikes when the temperature is between -5 to 25 degrees.

K. People tend to rent bikes when the visibility is between 300 to 1700.

L. The rentals were more in the morning and evening times. This is because people not having personal vehicles, commuting to offices and schools tend to rent bikes.

III. Training the model

A. Assigning the dependent and independent variables.

B. Splitting the model into train and test sets.

C. Transforming data using min-max scalar.

D. Fitting linear regression on the train set.

E. Getting the predicted dependent variable values from the model.

IV. Evaluating metrics of our model

A. Getting MSE, RMSE, R2-SCORE, and ADJUSTED-R2 SCORE for different models used.

a. MSE - the mean squared error or mean squared deviation of an estimator measures the average of the squares of the errors. b. RMSE - Root Mean Square Error (RMSE) is the standard deviation of the residuals (prediction errors). Residuals are a measure of how far from the regression line data points are.

c. R2-SCORE - R-squared (R2) is a statistical measure that represents the proportion of the variance for a dependent variable that's explained by an independent variable or variables in a regression model.

d. ADJUSTED-R2 SCORE - Adjusted R-squared is a modified version of R-squared that has been adjusted for the number of predictors in the model. The adjusted R-squared increases when the new term improves

the model more than would be expected by chance. It decreases when a predictor improves the model by less than expected.

B. Comparing the r2 score of all models used, to get the desired prediction.

Models used

Linear regression:

Regression models describe the relationship between variables by fitting a line to the observed data. Linear regression models use a straight line

---

Linear regression uses a linear approach to model the relationship between independent and dependent variables. In simple words, it's the best fit line drawn over the values of independent variables and dependent variables. In the case of a single variable, the formula is the same as a straight-line equation with an intercept and slope.

$$y\_pred=\beta_0+\beta_1 x$$

where

$$\beta_0 \text{ and } \beta_1$$

are intercept and slope respectively.

In case of multiple features, the formula translates into:

$$y\_pred=\beta_0+\beta_1 x_1+\beta_2 x_2+\beta_3 x_3+.....$$

where x_1,x_2,x_3 are the features values and

$$\beta_0,\beta_1,\beta_2.....$$

are weights assigned to each of the features These become the parameters that the algorithm tries to learn using Gradient descent?

Gradient descent is the process by which the algorithm tries to update the parameters using a loss function. The loss function is nothing but the difference between the actual values and predicted values(aka error or residuals). There are different types of loss functions but this is the simplest one. The loss function summed over all observation gives the cost functions.

The role of gradient descent is to update the parameters till the cost function is minimized i.e., a global minima is reached. It uses a hyperparameter 'alpha' that gives a weightage to the cost function and decides on how big the steps are to take. Alpha is called the learning rate. It is always necessary to keep an optimal value of alpha as high and low values of alpha might make the gradient descent overshoot or get stuck at local minima. There are also some basic assumptions that must be fulfilled before implementing this algorithm. They are:

1. No multi-collinearity in the dataset.
2. Independent variables should show a linear relationship with DV.
3. The residual mean should be 0 or close to 0.
4. There should be no heteroscedasticity i.e., the variance should be constant along the line of best fit.

Let us now implement our first model. We will be using Linear Regression from scikit library.

LASSO REGRESSION

Lasso regression is a type of linear regression that uses shrinkage. Shrinkage is where data values are shrunk towards a central point, like the mean. The lasso procedure encourages simple, sparse models (i.e. models with fewer parameters). This particular type of regression is well-suited for models showing high levels of multicollinearity or when you want to automate certain parts of model selection, like variable selection/parameter elimination. The acronym “LASSO” stands for Least Absolute Shrinkage and Selection Operator. Lasso solutions are quadratic programming problems, which are best solved with software (like Matlab). The goal of the algorithm is to minimize.

Ridge regression model:

Ridge regression is a model tuning method that is used to analyze any data that suffers from multicollinearity. This method performs L2

regularization. When the issue of multicollinearity occurs, least-squares are unbiased, and variances are large, this results in predicted values being far away from the actual values. The cost functions for ridge regression:

Min(||Y – X(theta)||^2 + λ||theta||^2)

Lambda is the penalty term. an alpha parameter denotes λ given here in the ridge function. So, by changing the values of alpha, we are controlling the penalty term. The higher the values of alpha, the bigger the penalty and therefore the magnitude of coefficients is reduced.

- It shrinks the parameters. Therefore, it is used to prevent multicollinearity.
- It reduces the model complexity by coefficient shrinkage

Elastic-net regularization model:

Elastic net linear regression uses the penalties from both the lasso and ridge techniques to regularize regression models. The technique combines both the lasso and ridge regression methods by learning from their shortcomings to improve the regularization of statistical models. The elastic net method improves lasso's limitations, i.e., where lasso takes a few samples for high dimensional data. The elastic net procedure provides the inclusion of "n" number of variables until saturation. If the variables are highly correlated groups, lasso tends to choose one variable from such groups and ignore the rest entirely. To eliminate the limitations found in lasso, the elastic net includes a quadratic expression (||β||2) in the penalty, which, when used in isolation, becomes ridge regression. The quadratic expression in the penalty elevates the loss function toward being convex. The elastic net draws on the best of both worlds – i.e., lasso and ridge regression.

DECISION TREE

Linear model trees combine linear models and decision trees to create a hybrid model that produces better predictions and leads to better insights than either model alone. A linear model tree is simply a decision tree with linear models at its nodes. This can be seen as a piecewise linear model with knots learned via a decision tree algorithm. LMTs can be used for regression problems (e.g. with linear regression models instead of population means) or classification problems (e.g. with logistic regression instead of population modes)

The random forest regression model

Random forests or random decision forests are an ensemble learning method for classification, regression, and other tasks that operates by constructing a multitude of decision trees at training time. For classification tasks, the output of the random forest is the class selected by most trees. For regression tasks, the mean or average prediction of the individual trees is returned.[1][2] Random decision forests correct for decision trees' habit of overfitting to their training set. Random forests generally outperform decision trees, but their accuracy is lower than gradient-boosted trees. However, data characteristics can affect their performance.

GRADIENT BOOSTING

Before proceeding to try the next models, let us try to tune some hyperparameters and see if the performance of our model improves.

Hyperparameter tuning is the process of choosing a set of optimal hyperparameters for a learning algorithm. A hyperparameter is a model

argument whose value is set before the learning process begins. The key to machine learning algorithms is hyperparameter tuning.

GridSearchCV helps to loop through predefined hyperparameters and fit the model on the training set. So, in the end, we can select the best parameters from the listed hyperparameters.

Challenges faced

- A huge amount of data needed to be dealt with while doing the project which is quite an important task and also even small inferences need to be kept in mind.
- As dataset was quite big enough which led to more computation time

CONCLUSION

During the time of our analysis, we initially did EDA on all the features of our dataset. We first analyzed our dependent variable, 'Rented Bike Count' and also transformed it. Then we analyzed the categorical variables and dropped the variable that had the majority of one class, we also analyzed numerical variables and found out the correlation, distribution, and their relationship with the dependent variable. We also removed some numerical features which had mostly 0 values and hot encoded the categorical variables.

Next, we implemented 7 machine learning algorithms Linear Regression, lasso, ridge, elastic net, decision tree, Random Forest, and XGBoost. We did hyperparameter

tuning to improve our model performance. The results of our evaluation are:

		Model	MAE	MSE	RMSE	R2_score	Adjusted R2
Training set	0	Linear regression	4.474	35.078	5.923	0.772	0.77
	1	Lasso regression	7.255	91.594	9.570	0.405	0.39
	2	Ridge regression	4.474	35.078	5.923	0.772	0.77
	3	Elastic net regression	5.792	57.574	7.588	0.626	0.62
	4	Dicision tree regression	3.209	23.746	4.873	0.846	0.84
	5	Random forest regression	0.801	1.598	1.264	0.990	0.99
	6	Gradient boosting regression	3.269	18.648	4.318	0.879	0.88
	7	Gradient Boosting gridsearchcv	1.849	7.455	2.730	0.952	0.95
Test set	0	Linear regression	4.410	33.275	5.768	0.789	0.78
	1	Lasso regression	7.456	96.775	9.837	0.387	0.37
	2	Ridge regression	4.410	33.277	5.769	0.789	0.78
	3	Elastic net regression Test	5.874	59.451	7.710	0.624	0.62
	4	Dicision tree regression	4.075	35.276	5.939	0.777	0.77
	5	Dicision tree regression	4.075	35.276	5.939	0.777	0.77
	6	Random forest regression	2.195	12.531	3.540	0.921	0.92
	7	Gradient boosting regression	3.493	21.289	4.614	0.865	0.86
	8	Gradient Boosting gridsearchcv	2.401	12.393	3.520	0.922	0.92

- No overfitting is seen.
- Random forest Regressor and Gradient Boosting gridsearchcv give the highest R2 score of 99% and 95% respectively for the Train Set and 92% for the Test set.
- Feature Importance value for Random Forest and Gradient Boost is different.
- We can deploy this model

However, this is not the ultimate end. As this data is time-dependent, the values for variables like temperature, wind speed, solar radiation, etc., will not always be consistent. Therefore, there will be scenarios where the model might not perform well. As Machine learning is an exponentially evolving field, we will have to be prepared for all contingencies and also keep checking our model from time to time. Therefore, having quality knowledge and keeping pace with the ever-evolving ML field would surely help one to stay a step ahead in the future.