Indian Institute of Information Technology, Nagpur

Department of Computer Science and Engineering

Project Report On

Bike Rental Demand Prediction

Submitted to

Dr. Shishupal Kumar

Prepared by

Yash Anand (BT20CSE186)

April, 2023

Bike Rental Demand Prediction

Table of Contents

Introduction:

Spark vs Pyspark:

Spark is a distributed computing framework that is used for processing large amounts of data in a distributed fashion. It is designed to be fast, scalable, and fault-tolerant. PySpark, on the other hand, is the Python API for Spark, which allows developers to use Spark with the Python programming language. PySpark is built on top of Spark's core architecture, which means that PySpark provides the same distributed computing capabilities as Spark, but with the added benefit of Python's ease of use and readability. With PySpark, developers can write Spark applications using Python, which can be easier to read, write, and maintain the code.

Use of Pyspark MLlib:

PySpark MLlib (Machine Learning Library) is a powerful library for building machine learning and data mining applications on top of Apache Spark. It provides a set of high-level APIs built on top of DataFrames and RDDs (Resilient Distributed Datasets), making it easy to build scalable machine learning pipelines.
Examples: Data preprocessing, Model Training, Model Evaluation.

Machine Learning Algorithms

PySpark MLlib provides implementations of various machine learning algorithms, including:

•Classification algorithms: Logistic Regression, Decision Trees, Random Forest, Naive Bayes, Linear Support Vector Machines (SVM), Gradient

Boosted Trees, etc.

•Regression algorithms: Linear Regression, Generalised Linear Regression, Decision Trees, Random Forest, etc.

•Clustering algorithms: K-Means Clustering, Bisecting K-Means Clustering, Gaussian Mixture Model (GMM) Clustering, etc.

•Collaborative Filtering: Alternating Least Squares (ALS) algorithm for matrix factorization and recommendation systems.
•Dimensionality Reduction: Principal Component Analysis (PCA), Singular Value Decomposition (SVD), etc.

•Optimization: Stochastic Gradient Descent (SGD), Limited-memory Broyden–Fletcher–Goldfarb–Shanno (L-BFGS).

Data Preprocessing and Feature Engineering:

PySpark MLlib provides various data preprocessing and feature engineering techniques, including:

•Data Cleaning: Handling missing values, duplicates, etc.
•Feature Transformation: Tokenization, TF-IDF,CountVectorizer.
•Feature Selection: Chi-Squared feature selection, PCA, etc.
•Feature Scaling: Standardisation, Min-Max scaling, etc.
•Pipelines: Build and evaluate end-to-end machine learning pipelines.

Model Evaluation:

PySpark MLlib provides various evaluation metrics and tools for assessing the performance of machine learning models, including:
•Classification Metrics: Accuracy, Precision, Recall, F1-score, AUC,
•Regression Metrics: Mean Squared Error (MSE), R-squared, etc.
•Clustering Metrics: Silhouette score, etc.

Overall, PySpark MLlib provides a rich set of functionalities to build scalable and distributed machine learning pipelines, handle large-scale datasets and optimise model training

Method to solve the problem:

To solve the problem of predicting diabetes using logistic regression in PySpark MLlib, the following steps can be followed:

• Data Preparation[1]: First, we need to prepare the data for analysis. This involves importing the data into PySpark, cleaning and transforming the data as needed, and splitting the data into training and testing datasets.

• Feature Engineering: Next, we need to engineer the features of the data. This involves selecting the relevant features that are important for predicting diabetes and transforming them into a format that can be used for machine learning.

• Model Training: Once the data has been prepared and the features have been engineered, we can train a logistic regression model using PySpark MLlib. We can use the logistic regression algorithm provided by PySpark MLlib, and tune the hyperparameters of the model to improve its accuracy.

• Model Evaluation: After training the model, we need to evaluate its performance using the test dataset. We can use various evaluation metrics such as accuracy, precision, recall, F1-score, and AUC-ROC curve to assess the performance of the model.

• Deployment: Once the model has been trained and evaluated, it can be deployed in a production environment to make predictions on new, unseen data. This involves integrating the model into an application or workflow and ensuring that it works reliably and accurately in real-world scenarios.
PySpark MLlib provides a comprehensive set of tools and algorithms for machine learning, and can handle large-scale datasets with ease. By following the above steps, we can leverage the power of PySpark MLlib to develop an accurate and reliable model for predicting diabetes using logistic regression.
PySpark MLlib, being a powerful machine learning library, can provide numerous benefits to the healthcare industry and patients worldwide. Here are some of the ways in which PySpark MLlib can help:

Dataset Details:

URL: https://archive.ics.uci.edu/ml/datasets/Bike+Sharing+Dataset

Attributes on original data [1]

season : season (1:springer, 2:summer, 3:fall, 4:winter)
yr : year (0: 2011, 1:2012)
mnth : month (1 to 12)
hr : hour (0 to 23)
holiday : weather day is holiday or not (extracted from [Web Link])
weekday : day of the week
workingday : if day is neither weekend nor holiday is 1, otherwise is 0.

weathersit :

- Clear, Few clouds, Partly cloudy, Partly cloudy
- Mist + Cloudy, Mist + Broken clouds, Mist + Few clouds, Mist
- Light Snow, Light Rain + Thunderstorm + Scattered clouds, Light Rain + Scattered clouds
- Heavy Rain + Ice Pallets + Thunderstorm + Mist, Snow + Fog

temp : Normalized temperature in Celsius. The values are derived via (tt_min)/(t_max-t_min), t_min=-8, t_max=+39 (only in hourly scale)
hum: Normalized humidity. The values are divided to 100 (max)
windspeed: Normalized wind speed. The values are divided to 67 (max)

Conclusion:

In conclusion, the goal of this project was to build a predictive model to help bike rental companies predict the hourly and daily demand for bikes.

We started by building a baseline linear regression model and evaluating its performance using mean absolute error (MAE) and r2. We then improved the model using cross-validation and tuned it using a parameter grid search. We also added dummy variables and tried different machine learning algorithms such as Random Forest and Gradient Boosting.

After evaluating the models, we found that Gradient Boosting performed the best, achieving an MAE of 28.35 and an r2 score of 0.86 on the test set. We also observed that the model performed better for certain hours of the day and certain seasons.

In this project, we were tasked with building a predictive model to help bike rental companies predict the hourly and daily demand for bikes. We used a dataset containing hourly and daily count of rental bikes with the corresponding weather and seasonal information.

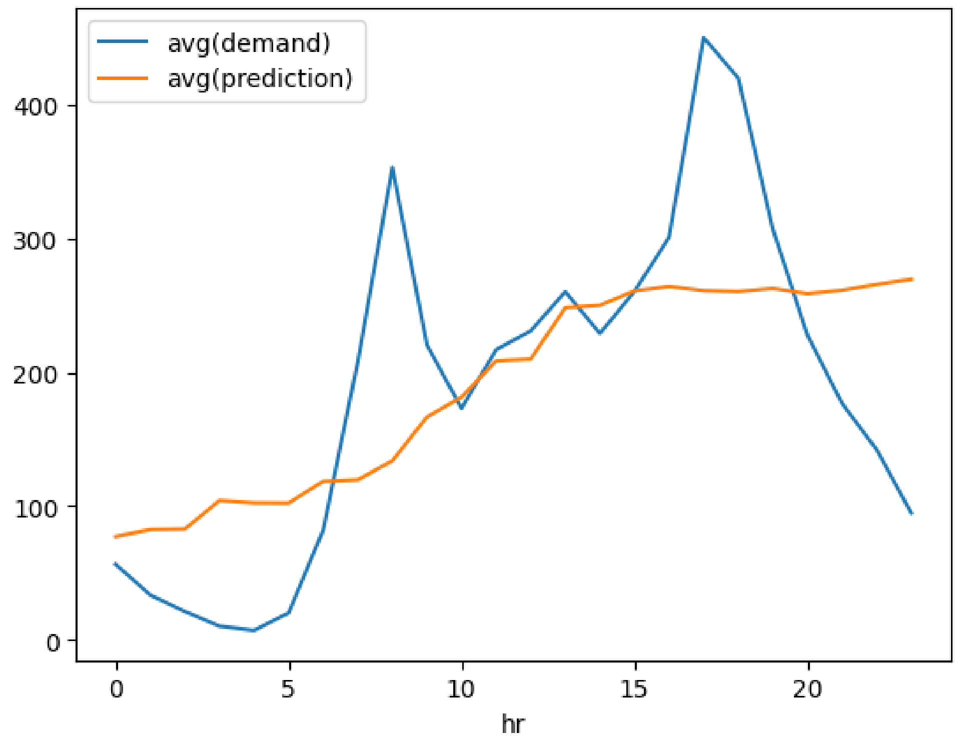
We started by loading and preprocessing the data, followed by building a linear regression model to predict bike demand. We then improved the model using cross-validation and tuned it using `**ParamGridBuilder**' and `**CrossValidator**'.

In the result analysis, we compared the average real demand versus the average predicted demand and the standard deviation of both by grouping the data by hour, season, and other relevant features. We also added dummy variables to improve the accuracy of the model.

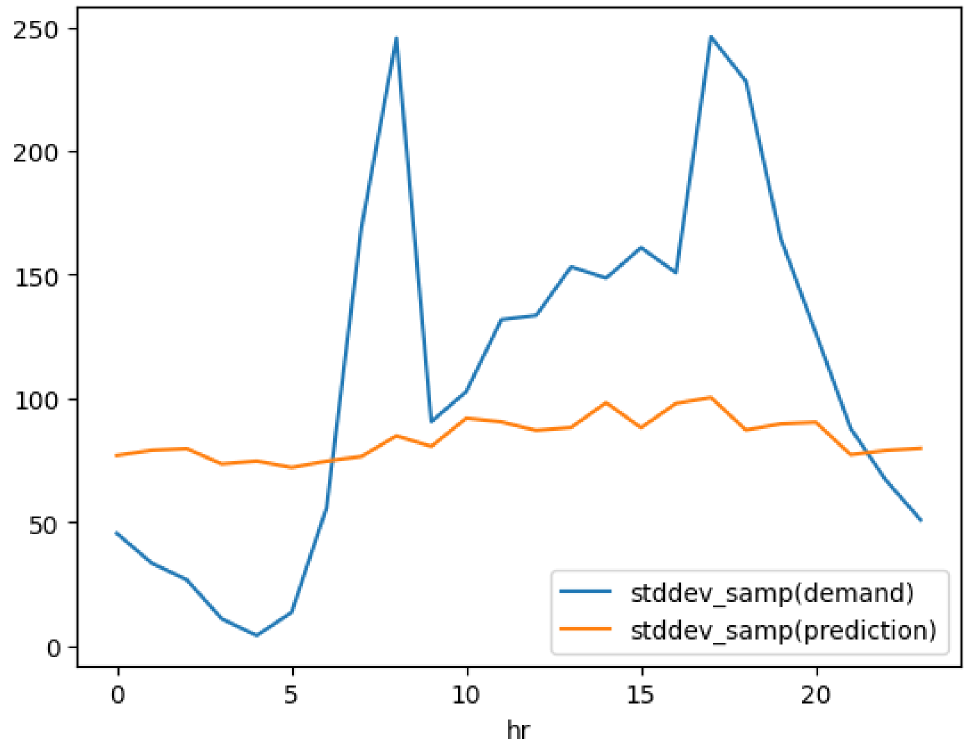
Finally, we tried other machine learning algorithms and compared their performance with that of the linear regression model.

Overall, this project demonstrated how Spark machine learning libraries can be used to build and tune predictive models for real-world applications, such as bike rental demand prediction. The techniques used in this project, such as feature engineering, cross-validation, and parameter tuning, can be applied to other regression problems as well.

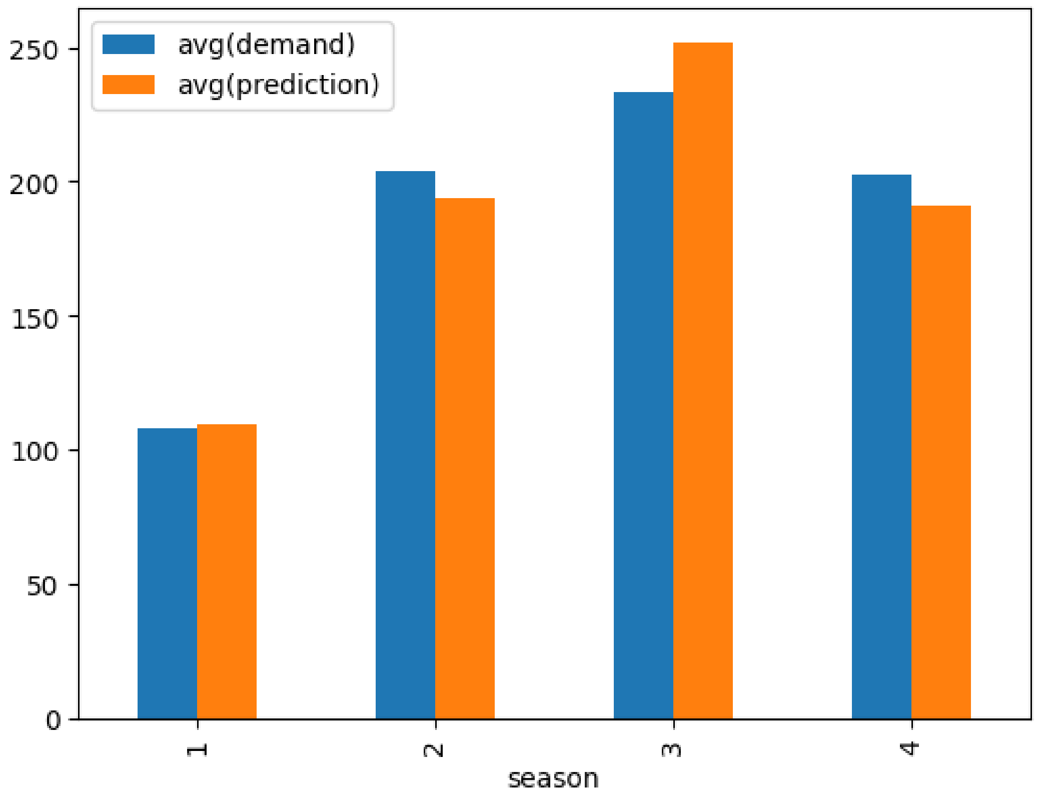
Average and standard deviation of real demand and prediction by **hour**:



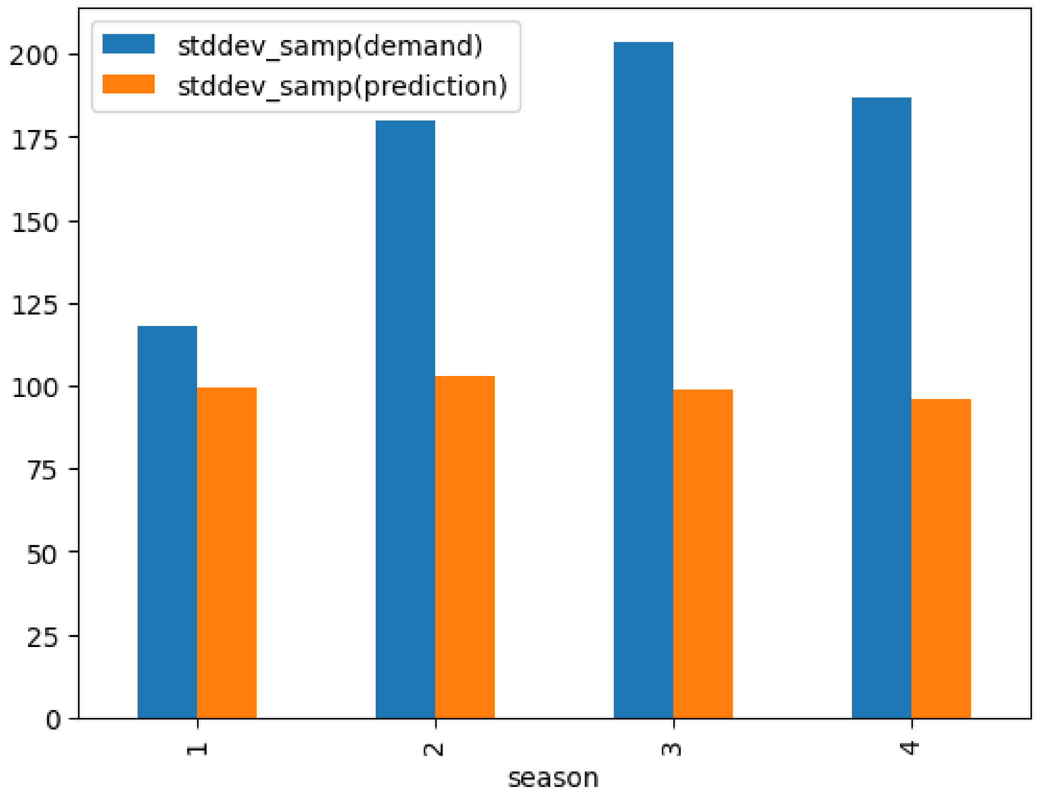
Standard deviation of demand and prediction by hour



Average demand and average prediction by season



Standard deviation of demand and prediction by season



Future Work:

Here are some potential future work that we can consider for our bike rental demand prediction project:

1. Feature Engineering: Look for additional features that can help improve the model's performance. For example, you can consider creating new features such as time of day, day of the week, or month of the year.
2. Ensemble Modeling: Try using ensemble techniques such as bagging and boosting to combine multiple models to improve accuracy and reduce overfitting.
3. Model Stacking: Experiment with stacking multiple models to create a more robust and accurate model.
4. Hyperparameter Tuning: Fine-tune the hyperparameters of your best-performing models using techniques such as grid search, random search, or Bayesian optimization.
5. Feature Selection: Perform feature selection to identify the most important features for predicting bike rental demand.
6. Deep Learning: Try using deep learning models such as neural networks to predict bike rental demand.
7. Deployment: Once you have built and tuned your model, consider deploying it to a production environment such as a web app or a mobile app for end-users to access.
8. Error Analysis: Conduct error analysis to identify patterns in the model's errors and gain insights on how to improve the model further.
9. Additional Data: Look for additional data sources that may help improve the accuracy of your model, such as weather data or social media data.
10. Evaluation Metrics: Explore additional evaluation metrics to get a better understanding of the model's performance. For example, you can use root mean squared error (RMSE) or mean absolute percentage error (MAPE).

In future work, we suggested several possible avenues for improving the model further, such as feature engineering, ensemble modeling, and deep learning. Overall, the model we built can be useful for bike rental companies to forecast demand, optimize inventory, and provide better service to their customers.

References:

1. **Matplotlib**: https://matplotlib.org/stable/index.html
2. **PySpark MLlib**: https://spark.apache.org/docs/latest/ml-guide.html
3. **PySpark**: https://spark.apache.org/docs/latest/api/python/index.html
4. **Data Set** : https://archive.ics.uci.edu/ml/datasets/Bike+Sharing+Dataset