## **Exercise 1 Report**

**Akash Shingha Bappy [2307938]**

**1. My Learning**: Today, from this exercise I learned about Spark which is a cluster based computer system that includes a library to process big data. I also learned some basic python commands to process big data such as creating a session, defining a data frame, opening data both locally and with url, saving/writing to a file, accessing data with sql like queries such as selecting, sorting, rounding different data. Finally I learned to visualize data as images with matplotlib and pandas library which I used later to visualize two different figures.

**2. Difficulties and Problems:**  Although the instructions were quite straightforward, I faced some issues while proceeding with the instruction. These are described below.

* I was unable to open the data from the url. Opening data from local storage was done successfully with *“df = spark.read.csv("file\_path")”*. But it failed to parse the data when the path was replaced with a url. I tried to solve it by taking help from the following [stackoverflow](https://stackoverflow.com/questions/69330177/pyspark3-read-file-from-https-url) and [databricks](https://community.databricks.com/t5/data-engineering/read-csv-directly-from-url-with-pyspark/td-p/12053) links. But that didn’t work correctly as the data was not well arranged and I could not resolve that.
* I have made a figure showing air\_temperature vs wind\_speed on a monthly basis. However, for this I needed to convert the dateTime from string to date format with the help of pandas.In the instruction there was a mention of “*pd.to\_numeric()”* to convert string to number, but it gave a blank figure. It was difficult for me as it was not mentioned in the instruction. However, with the help of [stackoverflow](https://stackoverflow.com/questions/25146121/extracting-just-month-and-year-separately-from-pandas-datetime-column) and [pandas documentation](https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.to_datetime.html) I was able to do it with “*dff['dateTime'] = pd.to\_datetime(dff['dateTime'])”*

**3. Figures**: figure 1 shows the wind speed with respect to air temperature. It can be seen that wind speed was higher around 10℃ and gradually decreased at higher or lower temperatures. However there are some outliers for 0℃. On the other hand, In figure 2 average wind speed is displayed corresponding to the months of different years. It is visible that it was the lowest in July which is summer and highest in December that is winter.

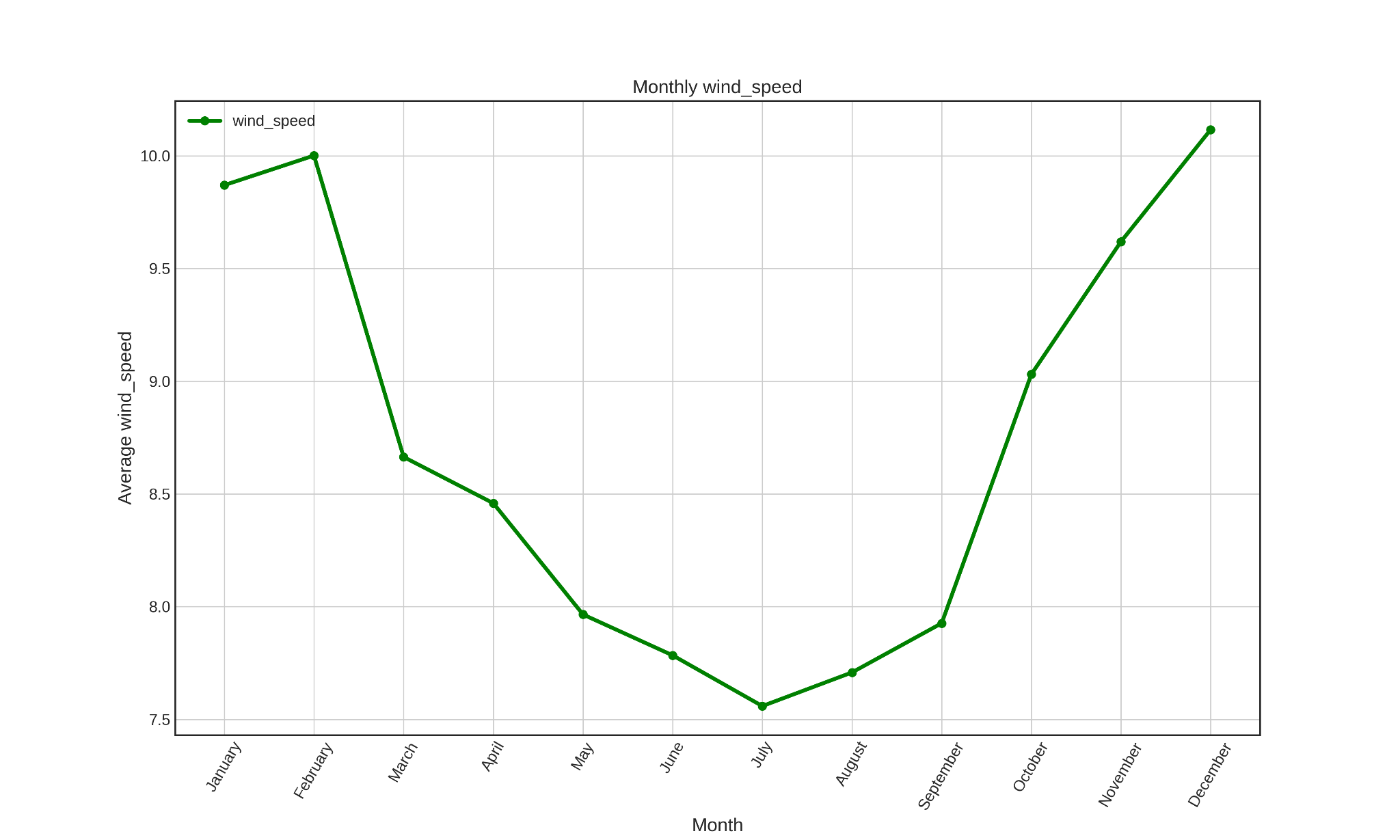
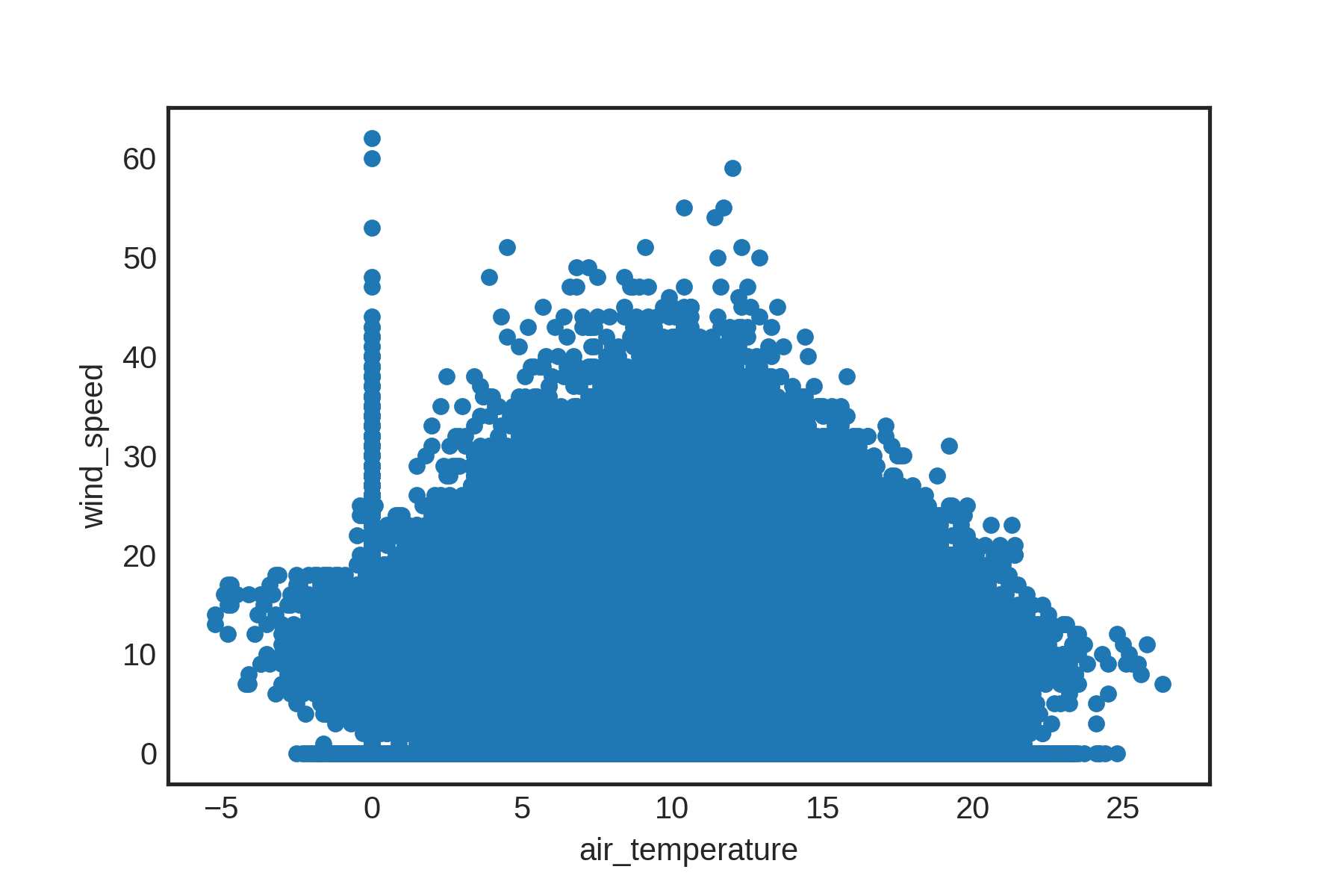


Figure 1: wind speed vs air temperature Figure 2: average wind speed vs Month

**Reference**:

1. <https://stackoverflow.com/questions/69330177/pyspark3-read-file-from-https-url>
2. <https://community.databricks.com/t5/data-engineering/read-csv-directly-from-url-with-pyspark/td-p/12053>
3. <https://stackoverflow.com/questions/25146121/extracting-just-month-and-year-separately-from-pandas-datetime-column>
4. <https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.to_datetime.html>

## **Exercise 2 Report**

**Akash Shingha Bappy [2307938]**

1. **My Learnings:**

- Structured Streaming is a high-level streaming API provided by Spark, introduced in Spark 2.2, that offers higher-level optimizations, event-time processing, and support for continuous processing.

- It perceives data streaming as continuously updating tables, making stream processing very similar to batch processing.

- Basic terminology includes Input Source, Sink, and Output Mode, each with several built-in options.

- Triggers define the timing of streaming data processing, including options like unspecified, fixed interval micro-batches, one-time micro-batch, and continuous with a fixed checkpoint interval.

- Event-Time Processing allows for processing based on the time embedded in the data itself, which is naturally expressed in the structured streaming model.

- Spark provides support for watermarking, allowing users to specify the threshold of late data and enabling the engine to clean up old state accordingly.

- Structured Streaming APIs include readStream and writeStream operations for reading and writing streaming data.

- Basic operations like filtering can be applied to streaming data similarly to batch data.

1. **Difference between methods and operations provided for structured streaming using SparkSession and Spark StreamingContext:**

- `SparkSession` is typically used for batch processing and structured streaming applications. It provides a unified entry point for reading data, executing SQL queries, and performing operations on DataFrames and Datasets.

- `SparkStreamingContext`, on the other hand, is specifically designed for handling real-time data streams. It provides functionality for creating DStreams (Discretized Streams), which represent a continuous stream of data.

- While both `SparkSession` and `SparkStreamingContext` can be used to work with structured streaming, their primary focus and APIs differ. `SparkSession` is more oriented towards batch and structured streaming, offering DataFrame-based APIs, while `SparkStreamingContext` is focused on handling real-time data streams using DStreams.