ISTANBUL AYDIN UNIVERSITY

Engineering Faculty

Department of Computer Engineering

SEN431- DATA MINING

Prof. Dr. Zafer ASLAN

Öykü AKYOL B2005.010064

Su YILMAZ B2005.010048

Ali İŞBİLİR B2005.010024

Arda AYGÜN B2005.010026

Eyüp DEMİRBİLEK B2005.010069

Content

Abstract 3

1 Introduction ..... ……3

2 Literature Review .....……4

3 Methodology 5

3.1 Data Preprocessing 5

3.2 Descriptive Statistics 11

3.3 Time Series Analysis 16

3.4 Clustering……18

3.5 Classification……...21

3.6 Future Predictions and Simulations…….22

4 Result and Visualization 24

4.1 Wind Speed Trends 24

4.2 Wind Direction Trends 25

5 Conclusion 26

6 References 26

Abstract

Renewable energy is of critical importance in shaping the future technology and industry. This article discusses the current status and future of renewable energy sources in the world energy industry. A literature review on the subject is included in the documentation. In addition, an analysis study on the potential of a specific location to produce wind energy based on weather data is included. Wind speed is of critical importance for wind energy generating infrastructure. The analyses conducted are prepared in accordance with the principles of data mining.

Keywords : energy , renewable energy , future , wind energy , wind speed

1 Introduction

Energy is a basic need for the continuation of the economic and social life of today's societies. The increasing energy demand in direct proportion to the increasing human population in the world, technological developments and growing industry is still primarily met by fossil fuels. The fact that fossil fuels (coal, oil, natural gas, etc.) cannot be used again after being used once and that the emissions they emit cause climate change by creating a greenhouse effect in the atmosphere is a concern for the future. The fact that these fuels provide energy quickly and in large quantities is an important reason for their continued use. The insufficient reserves of existing resources and the unstoppable energy demand in the world make it inevitable for countries to search for new resources. The most potential candidates that can replace these fuels, which can take centuries to form and can be consumed instantly, are renewable energy sources. Renewable energy sources (solar energy, biomass energy, hydroelectric energy, etc.) are the envisaged energy sources of the future with their reusability, sustainability, accessibility and most importantly, environmental friendliness. Since non-renewable energy sources have been primarily used in industry for many years, a rapid transition is not possible at once. Authorities in the world energy sector predict that 60 percent of the world's energy needs will be met by renewable energy sources by 2050. Building smart energy cities is currently among the targeted goals. However, studies on this subject are carried out by a small number of countries. The geopolitical, technological and economic inadequacies of countries can limit studies in this field. The inequality in the global energy race is clearly emphasized by experts. European Union countries such as Denmark and Germany are leading wind energy infrastructure studies. In the Asian continent, China and India are continuing studies in the field of solar and wind energy. Overseas countries such as the United States, Canada and Brazil have focused on infrastructure studies and integrations of more diverse renewable energy sources. Middle Eastern and African countries are behind in the global race due to infrastructure difficulties. The acceleration of global change in the field of renewable technology is promising for the future.

2. Literature Review

Most energy sources such as wind energy are dependent on geopolitical climatic conditions. Therefore, optimization methods should be used in the production, planning and control stages of energy. The increase in demand for energy worldwide makes the expansion of distribution networks important.

There are some multi-criteria decision-making methods applied to renewable energy problems in the literature (Pohekar and Ramachandran, 2004; Wang et al., 2009). Niknam and Firouzi (2009) showed that the method performed with the Nelder-Mead Simplex and PSO hybrid gives better results than other population-based methods. Niknam et al. (2010) proposed fuzzy adaptive PSO to solve the optimal operation management of distribution networks containing fuel cell power plants, since it gives better results compared to GA, PSO, KKA and Tabu Search (TA).

There are many studies showing the potential of wind energy in the world. The trends in wind power include the growing small-scale grid-connected turbines and a much wider variety of new wind projects in the world.

Li et al. (2010) applied the Bayesian method to long-term wind speed distributions. Zhao et al. (2009) used GA to determine the key specifications of inputs in the electrical system design of wind farms in terms of production cost and system reliability.

Wen et al. (2015) proposed a multi-objective hybrid PSO approach that takes into account the uncertainties in wind energy production to locate and size storage units in order to minimize the energy system cost and improve the system voltage level. In researches related to wind power, the important issue is the design of wind turbines and wind farm layout in the optimal wind farm design. Companies are trying to improve the average turbine sizes and technologies.

Benini and Toffolo (2002) used the multi-objective evolutionary approach in geometric parameter optimization to find the rotor configuration of horizontal axis wind turbines that will find the best performance in total energy production per square meter of the wind farm.

Kusiak and Zheng (2010) combined data mining and evolutionary computing to optimize the power produced by wind turbines. Other authors have proposed a decision analysis technique that includes a mixed integer nonlinear programming model for determining the optimum capacity by considering the uncertainties in the calculations originating from the wind speed distribution and power-speed characteristics (Kongnam et al., 2009).

Zakariazadeh et al. (2014) proposed a stochastic multi-objective economic /environmental/operational method for scheduling energy and reserves in a smart distribution system with high wind power effect.

Many wind speed estimation algorithms have been proposed in the literature to increase the estimation accuracy (Guo et al., 2011; Ren et al., 2014).

One of the most important problems in wind engineering is to estimate the output data of wind turbines that depend on the wind speed and system values. This can be explained as the reason why researchers use fuzzy logic modeling in wind turbine power curve estimation (Üstüntaş and Şahin, 2008).

Grady et al. (2005) presented a GA to determine the optimum placement of wind turbines for maximum generation capacity by limiting the number of installed turbines. Gebraad et al. (2016) presented a wind farm control strategy that optimizes the yaw settings of wind turbines for improved energy production of the wind farm.

3. Methodology

In this part of our study, first of all, data preprocessing steps were performed on the data set; missing values were checked and outliers were analyzed. The basic characteristics of the data were analyzed and visualized with descriptive statistics. Time series analysis was used to examine the change in wind speed over time, and then clustering and classification methods were used to group and categorize the data. Finally, simulations are applied to predict future wind speeds.

Jupyter Notebook and Python language were used during the study.

Used libraries are: Pandas, Numpy, Matplotlib, Seaborn, Scikit-learn (sklearn) and Sci-py

3.1 Data Pre-Processing

- At this stage, we first transferred our data to the Jupyter Notebook environment. After including the necessary libraries in the environment, we checked our data.

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

```
df = pd.read_csv('data.csv')
```

```
print(df.head())
```

```
  Unnamed: 0 Unnamed: 1 Unnamed: 2     Unnamed: 3
0       Year      Month        Day  SILIFKE_speed
1       2020          1          1            1.8
2       2020          1          2            1.6
3       2020          1          3            1.6
4       2020          1          4            2.1
```

- Whether there were empty rows, the type of data, column names and headings of the data were edited. A 'Date' column was created by combining the year, month and day columns in the dataset. This column was organized for time series analysis.

```
print(df.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1393 entries, 0 to 1392
Data columns (total 4 columns):
 #   Column      Non-Null Count  Dtype 
---  ------      --------------  ----- 
 0   Unnamed: 0  1393 non-null   object
 1   Unnamed: 1  1393 non-null   object
 2   Unnamed: 2  1393 non-null   object
 3   Unnamed: 3  1393 non-null   object
dtypes: object(4)
memory usage: 43.7+ KB
None
```

```
print(df.isnull().sum())
```

```
Unnamed: 0    0
Unnamed: 1    0
Unnamed: 2    0
Unnamed: 3    0
dtype: int64
```

```
# Veri setindeki sütun adlarını kontrol et
print(df.columns)
```

```
Index(['Unnamed: 0', 'Unnamed: 1', 'Unnamed: 2', 'Unnamed: 3'], dtype='object')
```

```
df.columns = ['Year', 'Month', 'Day', 'Silifke_speed']  # Başlıkları burada düzenle
```

```
print(df.columns)
```

```
Index(['Year', 'Month', 'Day', 'Silifke_speed'], dtype='object')
```

- Missing or non-numeric values in the data set were examined and necessary transformations were made

```
# Sayısal olmayan değerleri kontrol et
non_numeric_values = df['Silifke_speed'][pd.to_numeric(df['Silifke_speed'], errors='coerce').isna()]
print(non_numeric_values)
```

```
0    SILIFKE_speed
Name: Silifke_speed, dtype: object
```

```
df['Silifke_speed'] = pd.to_numeric(df['Silifke_speed'], errors='coerce')
```

```
print(df['Silifke_speed'].dtype)
```

```
float64
```

```
df = df.dropna(subset=['Silifke_speed'])
```

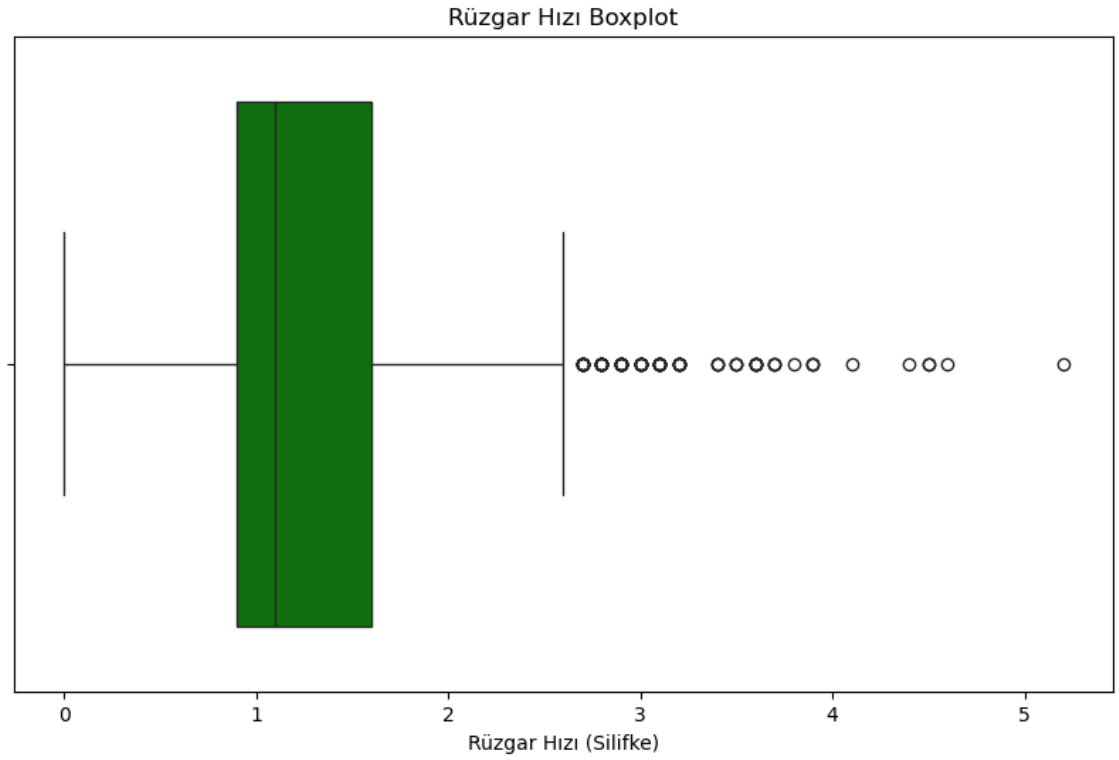
```
print(df['Silifke_speed'].head())
```

```
1    1.8
2    1.6
3    1.6
4    2.1
5    1.9
Name: Silifke_speed, dtype: float64
```

- The boxplot method was used to detect outliers. It is a visualization method used to show the central tendency, spread and possible outliers of a data set.

```python
import matplotlib.pyplot as plt
import seaborn as sns
```

```python
# Boxplot çizimi
plt.figure(figsize=(10,6))
sns.boxplot(x=df['Silifke_speed'], color='green')
plt.title('Rüzgar Hızı Boxplot')
plt.xlabel('Rüzgar Hızı (Silifke)')
plt.show()
```



- Normalization and standardization processes were performed to make the data more suitable for classification and clustering steps

```python
from sklearn.preprocessing import StandardScaler

# Standardizasyon işlemi
scaler = StandardScaler()
df['Silifke_speed_standardized'] = scaler.fit_transform(df[['Silifke_speed']])
```

```
from scipy import stats
```

```
# Z-skoru hesaplayın
z_scores = stats.zscore(df['Silifke_speed'])

# Aykırı değerler Z-skoru 3'ten büyük olanlardır
outliers = df[abs(z_scores) > 3]
print(outliers)
```

```
      Year Month Day  Silifke_speed       Date
10    2020     1  10            3.6 2020-01-10
22    2020     1  22            4.1 2020-01-22
24    2020     1  24            3.5 2020-01-24
73    2020     3  13            3.8 2020-03-13
77    2020     3  17            4.6 2020-03-17
78    2020     3  18            5.2 2020-03-18
385   2021     1  19            3.5 2021-01-19
386   2021     1  20            3.9 2021-01-20
413   2021     2  16            3.5 2021-02-16
414   2021     2  17            4.5 2021-02-17
437   2021     3  12            3.6 2021-03-12
466   2021     4  10            3.9 2021-04-10
681   2021    11  11            3.9 2021-11-11
723   2021    12  23            3.6 2021-12-23
745   2022     1  14            4.5 2022-01-14
750   2022     1  19            3.7 2022-01-19
758   2022     1  27            4.4 2022-01-27
809   2022     3  19            3.6 2022-03-19
1080  2022    12  15            3.6 2022-12-15
1129  2023     2   2            3.7 2023-02-02
1130  2023     2   3            3.6 2023-02-03
1135  2023     2   8            3.7 2023-02-08
```

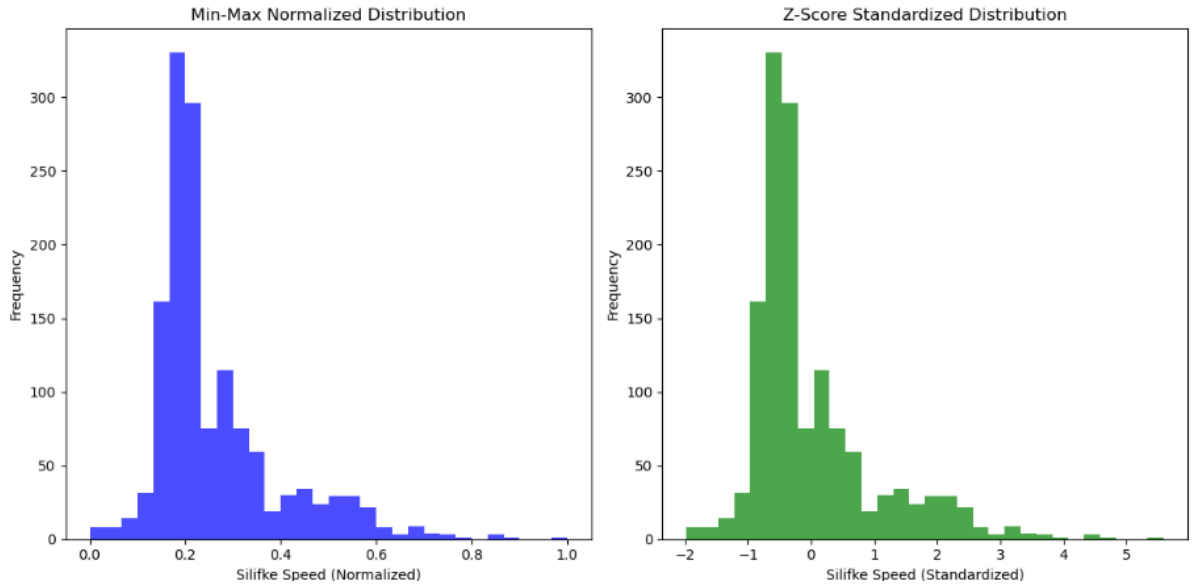
```
Q1 = df['Silifke_speed'].quantile(0.25)
Q3 = df['Silifke_speed'].quantile(0.75)
IQR = Q3 - Q1

# Aykırı değerleri tanımlama
lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR

outliers = df[(df['Silifke_speed'] < lower_bound) | (df['Silifke_speed'] > upper_bound)]
print(outliers)
```

```
      Year Month Day  Silifke_speed       Date
6     2020     1   6            2.8 2020-01-06
9     2020     1   9            2.8 2020-01-09
10    2020     1  10            3.6 2020-01-10
18    2020     1  18            3.0 2020-01-18
19    2020     1  19            3.0 2020-01-19
...    ...   ...  ..            ...        ...
1167  2023     3  12            2.8 2023-03-12
1216  2023     4  30            2.9 2023-04-30
1233  2023     5  17            2.7 2023-05-17
1246  2023     5  30            3.0 2023-05-30
1282  2023     7   5            3.1 2023-07-05

[96 rows x 5 columns]
```

```python
from sklearn.preprocessing import MinMaxScaler

# Min-Max Normalizasyonu
scaler = MinMaxScaler()
df['Silifke_speed_normalized'] = scaler.fit_transform(df[['Silifke_speed']])

# Yeni normalleştirilmiş veriyi gör
print(df[['Silifke_speed', 'Silifke_speed_normalized']].head())
```

```
   Silifke_speed  Silifke_speed_normalized
1            1.8                  0.346154
2            1.6                  0.307692
3            1.6                  0.307692
4            2.1                  0.403846
5            1.9                  0.365385
```

```python
import matplotlib.pyplot as plt

# Normalizasyon ve Standardizasyon sonrasında histogram
plt.figure(figsize=(12, 6))

# Min-Max Normalizasyonu Histogramı
plt.subplot(1, 2, 1)
plt.hist(df['Silifke_speed_normalized'], bins=30, color='blue', alpha=0.7)
plt.title('Min-Max Normalized Distribution')
plt.xlabel('Silifke Speed (Normalized)')
plt.ylabel('Frequency')

# Z-Score Standardizasyonu Histogramı
plt.subplot(1, 2, 2)
plt.hist(df['Silifke_speed_standardized'], bins=30, color='green', alpha=0.7)
plt.title('Z-Score Standardized Distribution')
plt.xlabel('Silifke Speed (Standardized)')
plt.ylabel('Frequency')

plt.tight_layout()
plt.show()
```



3.2 Descriptive Statistics

- In this step, the basic statistical measures on the data set were calculated with the describe function.

```
print(df.describe())
```

```
       Unnamed: 0 Unnamed: 1 Unnamed: 2 Unnamed: 3
count        1393       1393       1393       1393
unique          5         13         32         45
top          2020          1         16          1
freq          366        124         46        185
```

count: Indicates the number of values.

unique: Indicates the number of unique values.

top: Indicates the most frequent value.

freq: Indicates the number of repetitions of the most frequent value.

- Then we calculated mode, median, mean and standard deviation values.

```
mod_value = df['Silifke_speed'].mode()[0]
print("Mod:", mod_value)
```

```
Mod: 1
```

```
# Medyan (Ortadaki değer)
median_value = df['Silifke_speed'].median()
print("Medyan:", median_value)
```

```
Medyan: 1.1
```

```
# Ortalama (Mean)
mean_value = df['Silifke_speed'].mean()
print("Ortalama (Mean):", mean_value)
```

```
Ortalama (Mean): 1.3579741379310344
```

```
# Standart Sapma (Standard Deviation)
std_dev_value = df['Silifke_speed'].std()
print("Standart Sapma:", std_dev_value)
```

```
Standart Sapma: 0.6874302720376131
```

Mode: It is the value that appears most frequently in a data set

Median: It is the middle value of a data set when the values are ordered. If the number of elements in the data set is even, the median is the average of the two middle values.

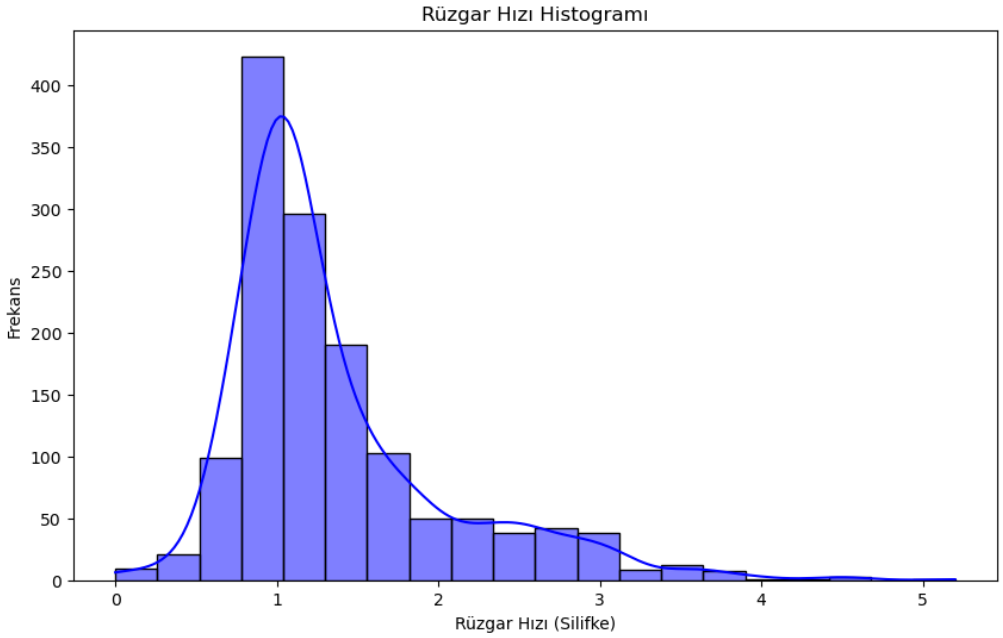
Mean: It is the average value of a data set, calculated by dividing the sum of all values by the number of values**.**

Standard Deviation: It measures the spread of data points around the mean, indicating how dispersed the values are.

- A histogram was created to visualize the distribution of wind speed data. A histogram is a type of bar chart that represents the frequency distribution of a dataset.
  It was used to observe the spread and central tendency of the wind speed values and to identify patterns or outliers in the data at this step.

```python
# Gerekli kütüphaneler
import matplotlib.pyplot as plt
import seaborn as sns
```

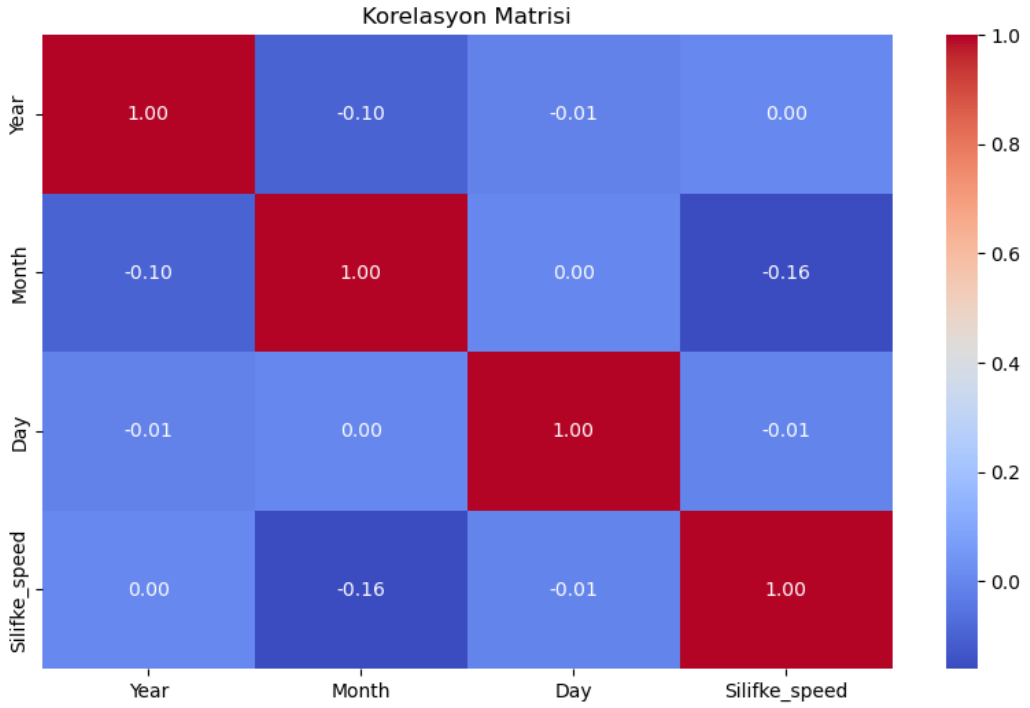
```python
# Histogram çizimi
plt.figure(figsize=(10,6))
sns.histplot(df['Silifke_speed'], bins=20, kde=True, color='blue')  # kde=True, dağılım çizgisi ekler
plt.title('Rüzgar Hızı Histogramı')
plt.xlabel('Rüzgar Hızı (Silifke)')
plt.ylabel('Frekans')
plt.show()
```



- A correlation matrix was created to examine the relationships between variables in the data set. The correlation matrix shows the relationship between each pair of variables in a numerical form. The values in this matrix range from -1 to 1: 1 indicates a perfect positive

relationship, -1 indicates a perfect negative relationship, and values close to 0 indicate that there is no relationship between the variables.
In this step, the relationships between wind speed (Silifke_speed) and other variables (e.g. year, month, day) were analyzed to determine which variables showed strong relationships with each other

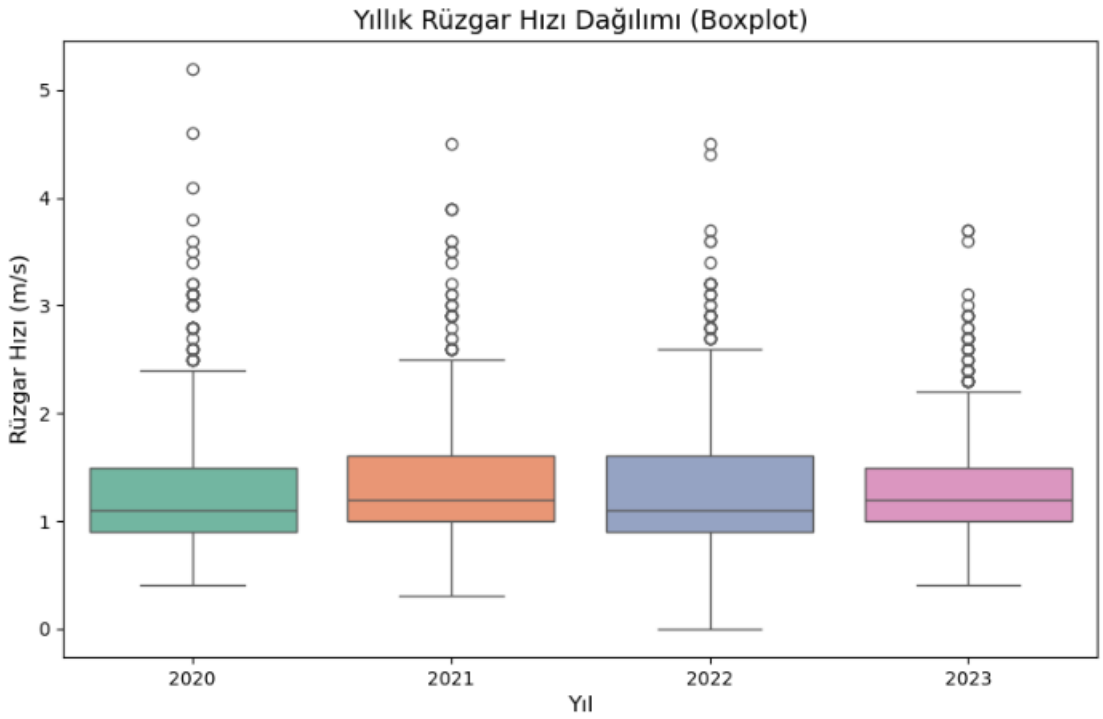
```
# Korelasyon Grafiği
corr_matrix = df.corr()
plt.figure(figsize=(10,6))
sns.heatmap(corr_matrix, annot=True, cmap='coolwarm', fmt='.2f')
plt.title('Korelasyon Matrisi')
plt.show()
```



- A boxplot was used to visualize the distribution of wind speed on an annual basis. These distributions help to understand annual trends, seasonal variations and possible data errors or outliers in the dataset, and are particularly useful to see in which years the data is more consistent or in which years wind speed is more erratic.

```
import matplotlib.pyplot as plt
import seaborn as sns
```

```
# Yıllık bazda boxplot oluşturma
plt.figure(figsize=(10, 6))
sns.boxplot(data=df, x='Year', y='Silifke_speed', palette='Set2')

# Başlık ve etiketler
plt.title('Yıllık Rüzgar Hızı Dağılımı (Boxplot)', fontsize=14)
plt.xlabel('Yıl', fontsize=12)
plt.ylabel('Rüzgar Hızı (m/s)', fontsize=12)

plt.show()
```



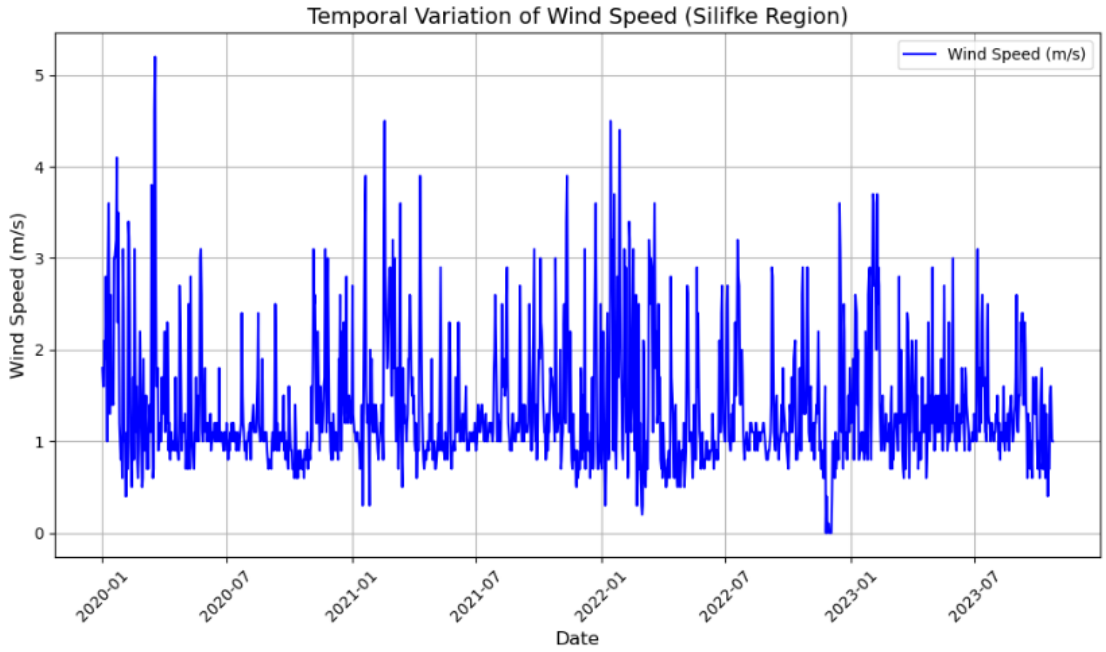
3.3 Time Series Analysis

- In the time series analysis step, the variation of the data over time was analyzed. Time series data were visualized and analyzed with graphs. In this way, seasonal patterns, possible peaks and long-term trends were identified.

- Temporal Variation Graphic was created to visualize the change of the data over time. This graph shows the relationship between time (day, month, year) and wind speed (Silifke_speed). With the help of this graph, the trends of wind speed over time, seasonal variations and fluctuations at certain periods during the year can be monitored. As a result, patterns or irregularities in the data set can be discovered more easily thanks to this visualization.

```
import matplotlib.pyplot as plt
```

```
# Zaman serisi grafiği çizme
plt.figure(figsize=(10, 6))
plt.plot(df['Date'], df['Silifke_speed'], label='Wind Speed (m/s)', color='b')

# Başlık ve etiketler
plt.title('Temporal Variation of Wind Speed (Silifke Region)', fontsize=14)
plt.xlabel('Date', fontsize=12)
plt.ylabel('Wind Speed (m/s)', fontsize=12)

# Dönemsel değişimi daha iyi görmek için x eksenindeki etiketleri döndürme
plt.xticks(rotation=45)

# Grafiği gösterme
plt.legend()
plt.grid(True)
plt.tight_layout()
plt.show()
```

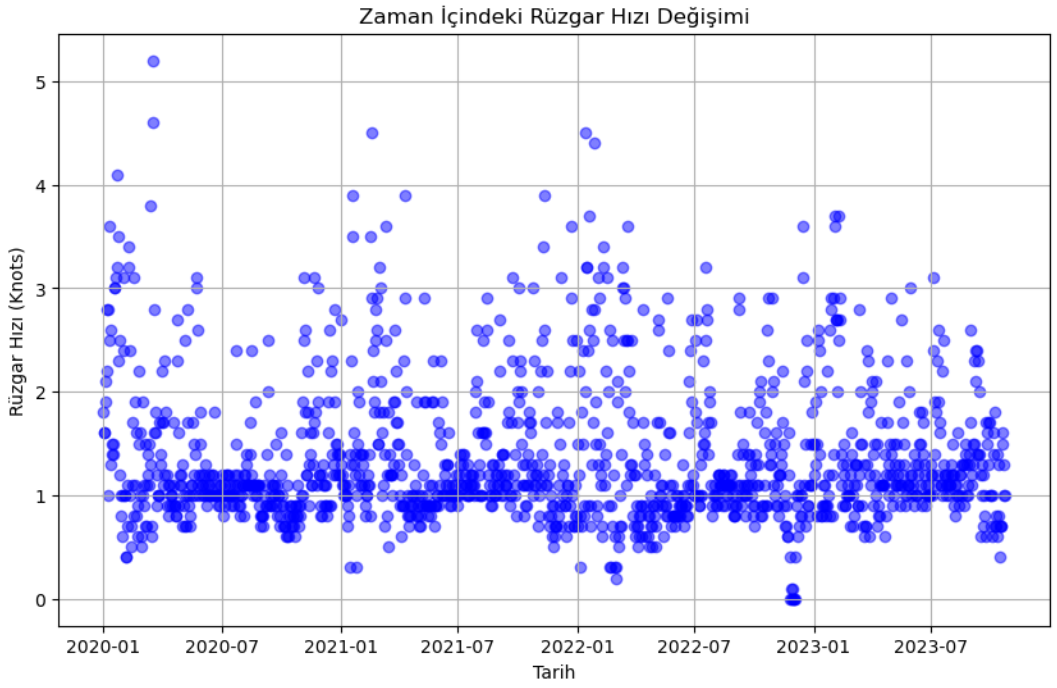


- Scatter Plot was used to visualize the relationship between wind speed and other variables (e.g. year, month, day). A scatter plot allows to examine linear or non-linear relationships between two variables. The X-axis represents one variable and the Y-axis represents the other variable. Each point represents a sample of the two variables.
  This method is used to quickly observe how the data are distributed and whether there is a relationship between the two variables. It is a very useful method to understand whether there are certain clusters, patterns or outliers in the data set.

```
# Zaman bilgisini oluşturma
df['Date'] = pd.to_datetime(df[['Year', 'Month', 'Day']])
```

```
import matplotlib.pyplot as plt
```

```
# Zaman ve rüzgar hızı arasındaki ilişkiyi gösteren scatter plot
plt.figure(figsize=(10,6))
plt.scatter(df['Date'], df['Silifke_speed'], alpha=0.5, color='blue')
plt.title('Zaman İçindeki Rüzgar Hızı Değişimi')
plt.xlabel('Tarih')
plt.ylabel('Rüzgar Hızı (Knots)')
plt.grid(True)
plt.show()
```



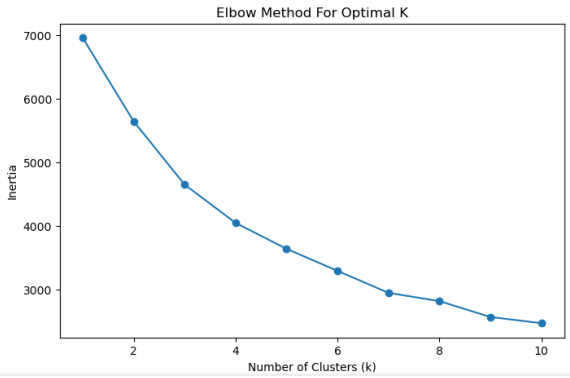
3.4 Clustering

```
file_path = 'Kitap1 veri.csv'  # Dosya adı
data = pd.read_csv(file_path)  # CSV dosyasını oku
print(data.head())
```

```
   Year  Month  Day  SILIFKE_Direction  SILIFKE_speed
0  2020      1    1                294            1.8
1  2020      1    2                298            1.6
2  2020      1    3                281            1.6
3  2020      1    4                282            2.1
4  2020      1    5                309            1.9
```

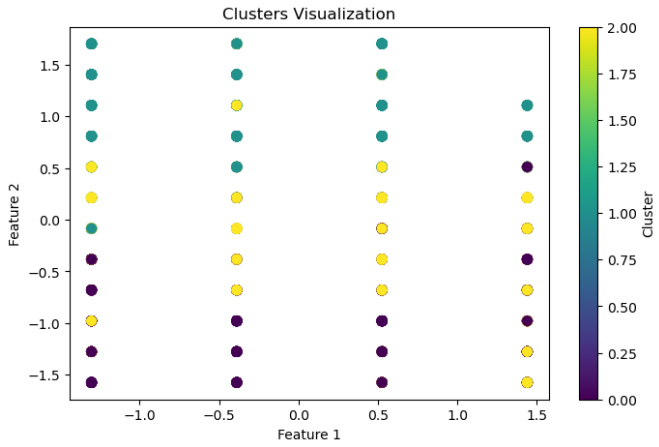
- **Elbow Method for K:**
  Elbow method; the sum of the squares of the distances of the points to the cluster center according to each K value is calculated. According to these values, a graph is drawn for each K value. The elbow point on the graph where the difference between the sums starts to decrease is determined as the most appropriate K value.

```
inertia = []
k_range = range(1, 11)

for k in k_range:
    kmeans = KMeans(n_clusters=k, random_state=42)
    kmeans.fit(scaled_data)
    inertia.append(kmeans.inertia_)

plt.figure(figsize=(8, 5))
plt.plot(k_range, inertia, marker='o')
plt.title('Elbow Method For Optimal K')
plt.xlabel('Number of Clusters (k)')
plt.ylabel('Inertia')
plt.show()
```



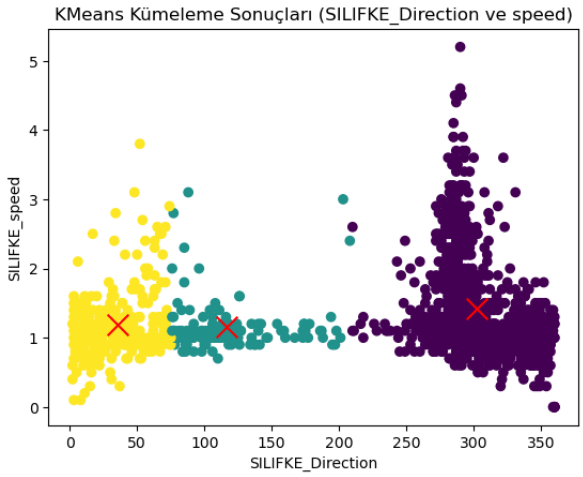
- **K-Means Clustering Visualization**
  The k-means algorithm is predicated on the insight that each data point should be proximate to the centre of its respective cluster.The algorithm is initiated by selecting k, the number of clusters to be identified in the data. The centres of these k clusters, designated as centroids, are then initialised.The algorithm then proceeds in two alternating parts: In the Reassign Points step, each data point is assigned to the cluster whose centroid is nearest to it. In the Update Centroids step, the location of each centroid is recalculated as the mean (center) of all the points assigned to its cluster. These steps are iterated until the centroids are static, or equivalently until the points cease to switch clusters.

```
import matplotlib.pyplot as plt

plt.figure(figsize=(8, 5))
plt.scatter(scaled_data[:, 0], scaled_data[:, 1], c=kmeans.labels_, cmap='viridis', s=50)
plt.title('Clusters Visualization')
plt.xlabel('Feature 1')
plt.ylabel('Feature 2')
plt.colorbar(label='Cluster')
plt.show()
```



- **K-Means Clustering Results**
  The success of K-Means clustering is measured by the compactness (closeness of points to the centre) and separation (distance between clusters) of clusters. Successful clustering is when clusters are clearly separated and the points within them are close to the centre. The accuracy of the model can be checked by examining the cluster centres and the distribution of data points.

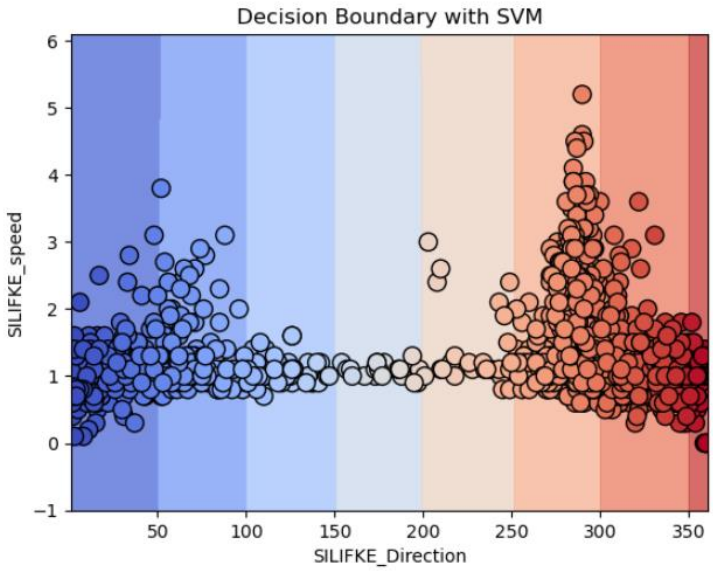
```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans

# Veriyi yükleme
file_path = 'Kitap1 veri.csv'
df = pd.read_csv(file_path)

print(df.columns)

X = df[["SILIFKE_Direction", "SILIFKE_speed"]].values

kmeans = KMeans(n_clusters=3)
kmeans.fit(X)

# Kümeleme sonuçlarını görselleştirme
plt.scatter(X[:, 0], X[:, 1], c=kmeans.labels_, cmap='viridis')
plt.scatter(kmeans.cluster_centers_[:, 0], kmeans.cluster_centers_[:, 1], s=200, c='red', marker='x')
plt.title('KMeans Kümeleme Sonuçları (SILIFKE_Direction ve speed)')
plt.xlabel('SILIFKE_Direction')
plt.ylabel('SILIFKE_speed')
plt.show()
```



3.5 Classification

- Classification represents an instance of a directed machine learning approach. The classification techniques in question provide assistance in making predictions about the category of the target values based on any given input. The spectrum of classification techniques encompasses binary classification and multi-class classification, among others. The categorisation of target values is contingent upon the number of classes incorporated within said values.

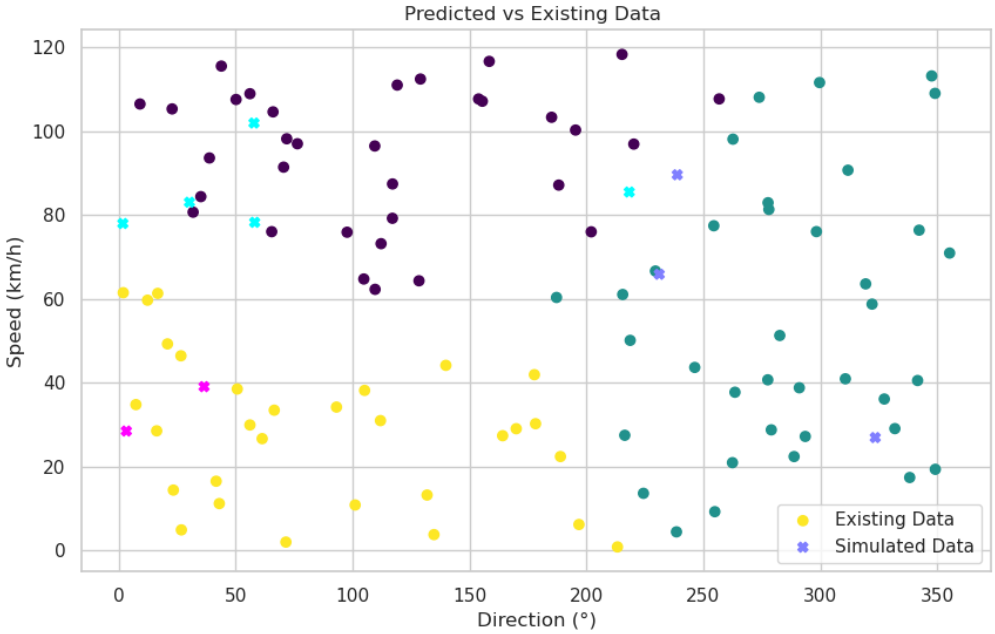
```
import matplotlib.pyplot as plt
import numpy as np
from sklearn.svm import SVC

# X özelliklerini (SILIFKE_Direction ve SILIFKE_speed) ve y hedef değişkenini (SILIFKE_Direction veya SILIFKE_speed) seçiyoruz
X = df[["SILIFKE_Direction", "SILIFKE_speed"]].values
y = df['SILIFKE_Direction']  # Hedef değişkeni olarak 'SILIFKE_Direction' seçtik

# SVC (Support Vector Classifier) ile model oluşturuluyor
model = SVC(kernel='linear')
model.fit(X, y)

# Karar sınırlarını çizme
x_min, x_max = X[:, 0].min() - 1, X[:, 0].max() + 1
y_min, y_max = X[:, 1].min() - 1, X[:, 1].max() + 1
xx, yy = np.meshgrid(np.arange(x_min, x_max, 0.1),
                     np.arange(y_min, y_max, 0.1))

Z = model.predict(np.c_[xx.ravel(), yy.ravel()])
Z = Z.reshape(xx.shape)

# Karar sınırlarını görselleştirme
plt.contourf(xx, yy, Z, alpha=0.75, cmap='coolwarm')
plt.scatter(X[:, 0], X[:, 1], c=y, edgecolors='k', marker='o', s=100, cmap='coolwarm')
plt.title('Decision Boundary with SVM')
plt.xlabel('SILIFKE_Direction')
plt.ylabel('SILIFKE_speed')
plt.show()
```



3.6 Future Predictions and Simulations

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split

# Verilerinizi yükleyin (örnek veri)
# df = pd.read_csv("your_data.csv")  # Burada kendi verinizi yüklemeniz gerekebilir.
# Ancak, burada örnek veriler oluşturuluyor.
np.random.seed(42)

# Silifke yön ve hız verileri simüle ediliyor
data = {
    'SILIFKE_Direction': np.random.uniform(0, 360, 100),
    'SILIFKE_speed': np.random.uniform(0, 120, 100)
}
df = pd.DataFrame(data)

# Veriyi ölçeklendiriyoruz (özellikle K-Means için gerekli)
scaler = StandardScaler()
df_scaled = scaler.fit_transform(df)

# K-Means uyguluyoruz
kmeans = KMeans(n_clusters=3, random_state=42)
df['Cluster'] = kmeans.fit_predict(df_scaled)

# Yeni simülasyon (geçici olarak 10 yeni veri noktası ekleyelim)
new_data = {
    'SILIFKE_Direction': np.random.uniform(0, 360, 10),
    'SILIFKE_speed': np.random.uniform(0, 120, 10)
}
new_df = pd.DataFrame(new_data)
```

```
# Yeni simülasyon (geçici olarak 10 yeni veri noktası ekleyelim)
new_data = {
    'SILIFKE_Direction': np.random.uniform(0, 360, 10),
    'SILIFKE_speed': np.random.uniform(0, 120, 10)
}
new_df = pd.DataFrame(new_data)

# Yeni veriyi ölçeklendiriyoruz
new_scaled = scaler.transform(new_df)

# Yeni verileri mevcut modelle tahmin ediyoruz
new_clusters = kmeans.predict(new_scaled)

# Simülasyon ve mevcut veriyi görselleştiriyoruz
plt.figure(figsize=(10, 6))
plt.scatter(df['SILIFKE_Direction'], df['SILIFKE_speed'], c=df['Cluster'], cmap='viridis', label='Existing Data')
plt.scatter(new_df['SILIFKE_Direction'], new_df['SILIFKE_speed'], c=new_clusters, marker='X', cmap='cool', label='Simulated Data')
plt.xlabel('Direction (°)')
plt.ylabel('Speed (km/h)')
plt.title('Predicted vs Existing Data ')
plt.legend()
plt.grid(True)
plt.show()
```



- This graph shows how the data resulting from the predictions made with the available data are grouped using the K-Means clustering algorithm. Actual data and data based on predictions are shown in a different format. The graph illustrates how new data points interact with existing clusters and how the algorithm can classify this new data. This type of visualisation helps to evaluate the predictive performance of the model on future data. By interpreting this model, information about future data such as wind direction and wind speed can be obtained.c

- The dataset contains wind-related data for Silifke, including measurements of wind speed (in m/s) and wind direction (in degrees). Based on this historical data, future simulations were conducted to predict wind behavior for the next 365 days. The simulations incorporate statistical trends and patterns observed in the historical data, generating realistic synthetic data for wind speed and direction.

- **Results and Visualizations**

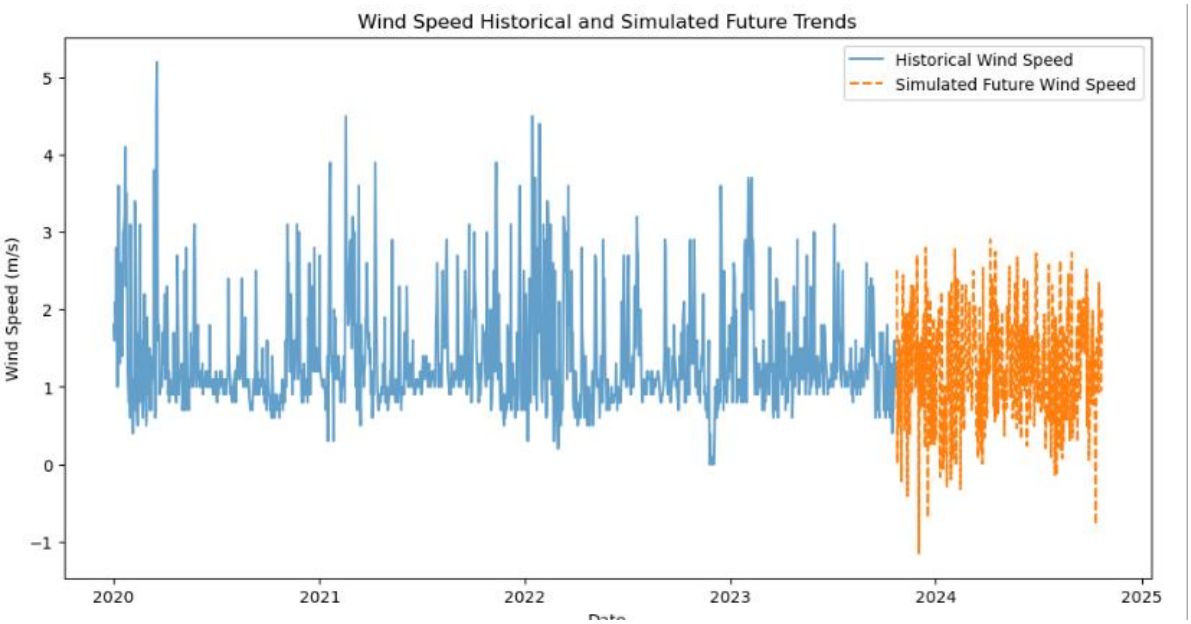
The following visualizations will be include wind speed trends and wind direction trends.

Code for these visualizations:

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression

# Prepare the data for simulation
data['Date'] = pd.to_datetime(data[['Year', 'Month', 'Day']])
data.set_index('Date', inplace=True)

# Extract relevant columns
wind_speed = data['SILIFKE_speed']
wind_direction = data['SILIFKE_Direction']

# Generate a linear regression model for wind speed
days_since_start = np.arange(len(wind_speed)).reshape(-1, 1)
model_speed = LinearRegression().fit(days_since_start, wind_speed)
speed_trend = model_speed.predict(days_since_start)

# Simulate future data (next 365 days)
future_days = np.arange(len(wind_speed), len(wind_speed) + 365).reshape(-1, 1)
future_speed_trend = model_speed.predict(future_days)
future_speed_simulated = future_speed_trend + np.random.normal(0, wind_speed.std(), len(future_days))

# Simulate wind direction as a periodic pattern
direction_mean = wind_direction.mean()
direction_std = wind_direction.std()
future_direction_simulated = (direction_mean + direction_std * np.sin(2 * np.pi * future_days.flatten() / 365)) % 360

# Create a future date range
future_dates = pd.date_range(start=wind_speed.index[-1] + pd.Timedelta(days=1), periods=365)

# Combine future data into a DataFrame
future_data = pd.DataFrame({
    'Date': future_dates,
    'Simulated_Wind_Speed': future_speed_simulated,
    'Simulated_Wind_Direction': future_direction_simulated
}).set_index('Date')

# Visualize the results
plt.figure(figsize=(12, 6))
plt.plot(wind_speed.index, wind_speed, label="Historical Wind Speed", alpha=0.7)
plt.plot(future_dates, future_speed_simulated, label="Simulated Future Wind Speed", linestyle='--')
plt.title("Wind Speed Historical and Simulated Future Trends")
plt.xlabel("Date")
plt.ylabel("Wind Speed (m/s)")
plt.legend()
plt.show()

plt.figure(figsize=(12, 6))
plt.plot(wind_direction.index, wind_direction, label="Historical Wind Direction", alpha=0.7)
plt.plot(future_dates, future_direction_simulated, label="Simulated Future Wind Direction", linestyle='--')
plt.title("Wind Direction Historical and Simulated Future Trends")
plt.xlabel("Date")
plt.ylabel("Wind Direction (degrees)")
plt.legend()
plt.show()
```

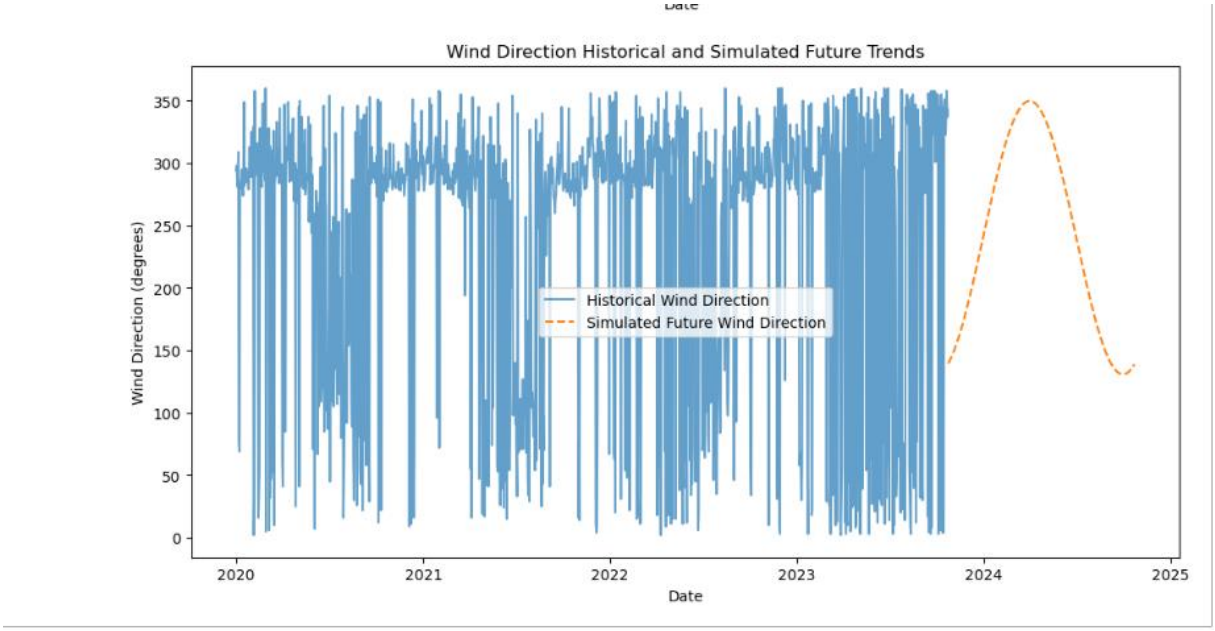
Wind Speed Trends:

- Historical wind speed data is plotted alongside simulated future wind speeds.
- The future data extends the historical trend linearly while adding random variability to replicate natural fluctuations.



Wind Direction Trends:

- Historical wind direction data exhibits periodic variability, which was captured in the simulation.
- Simulated wind directions follow a sinusoidal pattern, maintaining the cyclical characteristics observed in historical data.



4. Conclusion

In this study, first of all, examined that what renewable energy sources are and how they work. In particular, the focus was on wind energy and the potential, usage areas and advantages of this energy source were mentioned. It was emphasized that wind energy is an important resource in terms of environmental benefits and sustainability.

Afterwards, an in-depth literature review of studies on wind energy was conducted. Information on various studies, innovative technologies and the adaptability of wind energy to different regions and conditions was obtained. This review formed the basis of the study and demonstrated the existing body of knowledge.

Then, in the methodology section, the steps for analyzing the data were carried out sequentially. Techniques such as data preprocessing, descriptive statistical analysis, and time series analysis were used to examine the characteristics and trends of the data in more depth. At these stages, the necessary cleaning and normalization processes were performed to ensure the accuracy of the data and make it suitable for analysis. Again, visualizations were used to observe the relationships between the data and their changes over time.The results of the study were reinforced by using forecasts and simulations for the future

5. References

[1] Sayed, E. T., Olabi, A. G., Alami, A. H., Radwan, A., Mdallal, A., Rezk, A., & Abdelkareem, M. A. (2023). Renewable energy and energy storage systems. Energies, 16(3), 1415.

[2] Gross, R., Leach, M., & Bauen, A. (2003). Progress in renewable energy. Environment international, 29(1), 105-122.

[3] ERVURAL, B. Ç., ERVURAL, B., & EVREN, R. (2016). Enerjide Optimizasyon Uygulamaları: Bir Literatür Araştırması. Ege Academic Review, 16.

[4] Hassan, Q., Viktor, P., Al-Musawi, T. J., Ali, B. M., Algburi, S., Alzoubi, H. M., ... & Jaszczur, M. (2024). The renewable energy role in the global energy Transformations. Renewable Energy Focus, 48, 100545.

[5] ŞAHİN, B., & ŞAHİN, Y. (2024). Yenilenebilir Enerji Kaynakları ve Yeşil Finans Alanında Yapılan Çalışmaların Yazın Taraması. MAS Journal of Applied Sciences, 9(2), 414-426.