Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ __Информатика и системы управления_____

КАФЕДРА _Системы обработки информации и управления_____

РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

К НАУЧНО-ИССЛЕДОВАТЕЛЬСКОЙ РАБОТЕ

НА ТЕМУ:

____Анализ мощности ветровой установки__

Студент _ИУ5-32М___ _____ **Цапий В.С.**
(Группа) (Подпись, дата) (И.О.Фамилия)

Руководитель _____ **Гапанюк Ю.Е.**
(Подпись, дата) (И.О.Фамилия)

2020г.

1. Введение

Обработка и анализ данных занимают важную роль в проектах машинного обучения и нейронных сетей. От собранных данных зависит точность предсказания и общий анализ области. Предварительную обработку и чистку данных необходимо выполнить, прежде чем набор данных можно будет использовать для обучения модели. Необработанные данные зачастую искажены и ненадежны, имеют некорректный формат или имеют пропущенные значения в данных. Использование таких данных приводит к некорректным и неверным результатам. Это подразумевает первоначальное изучение набора данных, используемых для определения и планирования необходимой обработки.

2. Постановка задачи

1) Необходимо провести предварительную обработку собранных метеорологических данных за последние 15 лет. Данные собирались с метеорологической станции в Астраханской области.
2) Изучить датасет с открытыми данными выработанной электроэнергии ветровыми установками, провести анализ и построить предварительную модель.
3) На основе изученных ранее данных датасета, провести расчет теоретического значения выработанной электроэнергии для собранных метеорологических данных погоды.
4) В процессе предварительной обработки необходимо произвести:
    - Очистку данных
    - Преобразование данных
    - Уплотнение данных
    - Дискретизацию данных
    - Очистку текста

3. Предварительная обработка и очистка данных

Реальные данные собираются для последующей обработки из разных источников и процессов. Они могут содержать ошибки и повреждения, негативно влияющие на качество набора данных. Вот какими могут быть типичные проблемы с качеством данных:

- **Неполнота**: данные не содержат атрибутов, или в них пропущены значения.
- **Шум**: данные содержат ошибочные записи или выбросы.
- **Несогласованность**: данные содержат конфликтующие между собой записи или расхождения.

Качественные данные — это необходимое условие для создания качественных моделей прогнозирования. Чтобы избежать появления ситуации «мусор на входе, мусор на выходе» и повысить качество данных и, как следствие, эффективность модели, необходимо провести мониторинг работоспособности данных, как можно раньше обнаружить проблемы и решить, какие действия по предварительной обработке и очистке данных необходимы.

Главные задачи предварительной обработки данных:

- Очистка данных – заполнение отсутствующих значений, обнаружение шума данных и выбросов
- Преобразование данных – нормализация данных для уменьшения размеров и шума
- Уплотнение данных – создание выборки данных или атрибутов для упрощения обработки данных
- Дискретизация данных – преобразование непрерывных атрибутов в категориальные, чтобы проще было использовать некоторые методы машинного обучения

- Очистка текста – удаление внедренные символы, которые могут вызвать неправильное выравнивание данных, например, внедренные вкладки в файле данных с разделителями-табуляцией, новые строки, которые могу нарушить работу записей.

3.1 Поиск и выбор набора данных для построения моделей машинного обучения

В качестве набора данных я взял открытый датасет с данными о выработанной мощности электроэнергии ветровой установки в Турции (https://www.kaggle.com/berkerisen/wind-turbine-scada-dataset).
Данный датасет является основой для предварительного анализа и обработки данных для дальнейших собранных данных метеорологических установок.
Состав таблицы представляет:

- Date/Time – Дата и время ряда с 10-ти минутным интервалом
- ActivePower (kW) – Мощность вырабатываемая турбиной в данный момент времени
- Wind Speed (m/s) – Скорость ветра
- TheoreticalPowerCurve (KWh) – Теоретическая мощность вырабатываемая ветровой установкой
- Wind direction – Направление ветра

Постановка задачи:

Ввиду преимущества численного типа данных в выбранном наборе, было принято решение о выполнении поставленной задачи методом регрессии, так как подход с использованием классификации повлек бы к дополнительному этапу классифицирования данных в числовом формате.

3.2 Проведение разведочного анализа данных. Анализ и заполнение пропусков данных.

Выполним импорт библиотек, необходимых для дальнейшего анализа и обработки информации:

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

С помощью библиотеки Pandas, подключим набор данных:

data_wind = pd.read_csv(r"C:\Users\VTsapiy\Downloads\датасет для диплома\T1.csv")

Для ознакомления с данными из набора, отобразим первые строки из датасета:

data_wind.head()

	Date/Time	LV ActivePower (kW)	Wind Speed (m/s)	Theoretical_Power_Curve (KWh)	Wind Direction (°)
0	01 01 2018 00:00	380.047791	5.311336	416.328908	259.994904
1	01 01 2018 00:10	453.769196	5.672167	519.917511	268.641113
2	01 01 2018 00:20	306.376587	5.216037	390.900016	272.564789
3	01 01 2018 00:30	419.645905	5.659674	516.127569	271.258087
4	01 01 2018 00:40	380.650696	5.577941	491.702972	265.674286

Размер датасета можно посмотреть командой

data_wind.shape

```
(50530, 5)
```

Представленный набор содержит 50530 строк и 5 колонок.

Для более удобного отображения количества строк используем

total_count = data_wind.shape[0]

print("Всего строк {}".format(total_count))

```
Всего строк 50530
```

Посмотрим все колонки которые есть в нашем наборе данных:

data_wind.columns

```
Index(['Date/Time', 'LV ActivePower (kW)', 'Wind Speed (m/s)',
       'Theoretical_Power_Curve (KWh)'],
      dtype='object')
```

С типом переменных для набора данных можно посмотреть командой

data_wind.dtypes

```
Date/Time                       object
LV ActivePower (kW)            float64
```

```
Wind Speed (m/s)                   float64
Theoretical_Power_Curve (KWh)      float64
Wind Direction (°)                 float64
dtype: object
```

Все колонки, кроме Date/Time имеют формат float64, числовой формат с плавающей точкой.

Узнаем количество пробелов(пропусков) в нашем наборе

data_wind.isnull().sum()

```
Date/Time                          0
LV ActivePower (kW)                0
Wind Speed (m/s)                   0
Theoretical_Power_Curve (KWh)      0
Wind Direction (°)                 0
dtype: int64
```

Для того, чтобы узнать количество пустых значений

```
cat_cols = []
for col in data_wind.columns:
    # Количество пустых значений
    temp_null_count = data_wind[data_wind[col].isnull()].shape[0]
    dt = str(data_wind[col].dtype)
    if temp_null_count>0 and (dt=='object'):
        cat_cols.append(col)
        temp_perc = round((temp_null_count / total_count) * 100.0, 2)
        print('Колонка {}. Тип данных {}. Количество пустых значений {},
{}%.'.format(col, dt, temp_null_count, temp_perc))
```

Так же можно воспользоваться другим способом

```
for col in data_wind.columns:
    temp_null_count = data_wind[data_wind[col].isnull()].shape[0]
    print('{} - {}'.format(col, temp_null_count))
```

```
Date/Time - 0
LV ActivePower (kW) - 0
Wind Speed (m/s) - 0
Theoretical_Power_Curve (KWh) - 0
Wind Direction (°) - 0
```

Все данные не имеют пропусков, это свидетельствует о качестве данного датасета.

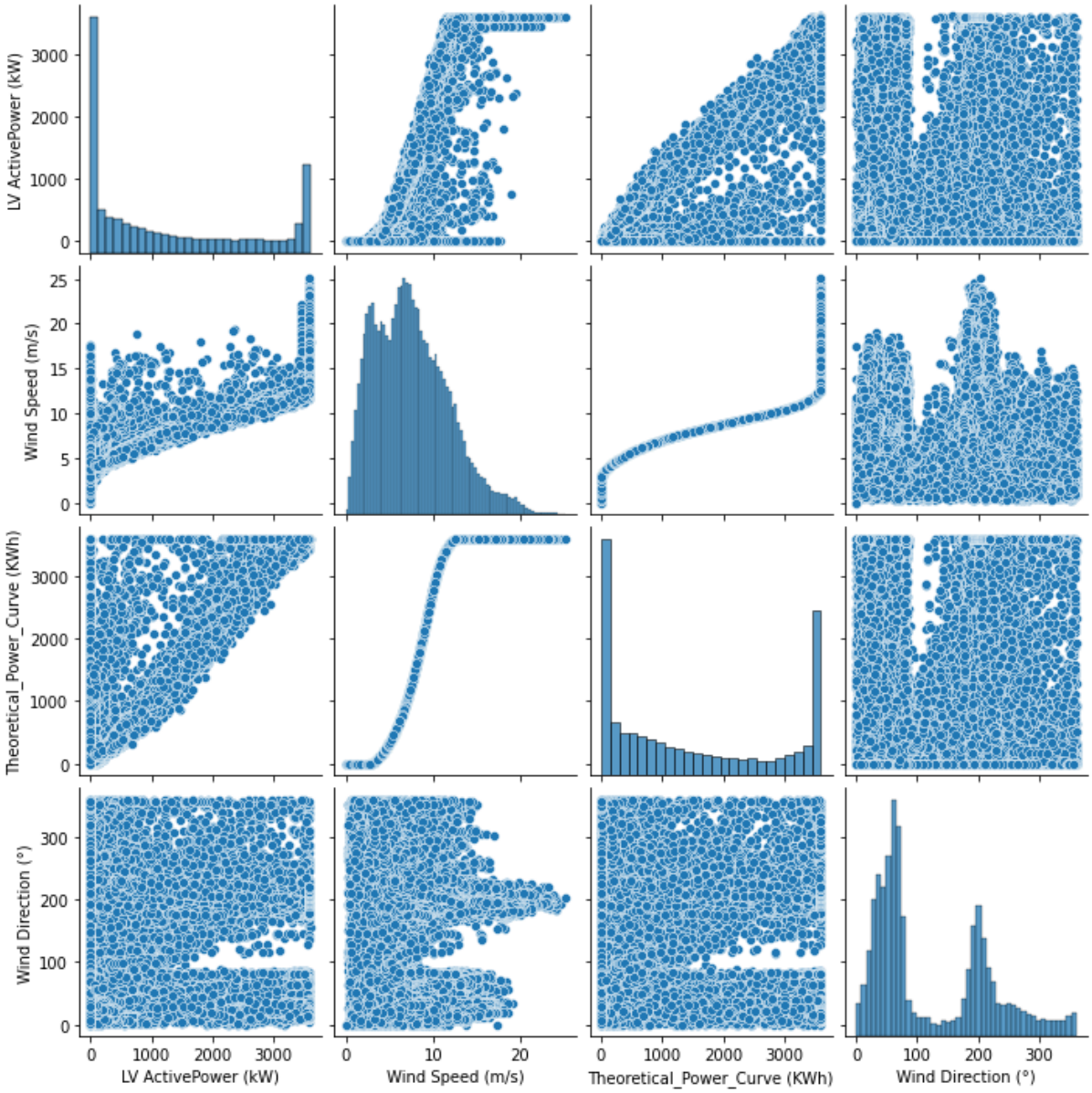
Исследовав пропущенные значения, познакомимся с описательными статистиками выбранного набора данных:

```
strippedCols = dict()
for name in data_wind.columns:
  strippedCols[name] = name.strip()
data_wind = data_wind.rename(strippedCols, axis='columns', errors='raise')
data_wind.describe()
```

	LV ActivePower (kW)	Wind Speed (m/s)	Theoretical_Power_Curve (KWh)	Wind Direction (°)
count	50530.000000	50530.000000	50530.000000	50530.000000
mean	1307.684332	7.557952	1492.175463	123.687559
std	1312.459242	4.227166	1368.018238	93.443736
min	-2.471405	0.000000	0.000000	0.000000
25%	50.677890	4.201395	161.328167	49.315437
50%	825.838074	7.104594	1063.776283	73.712978
75%	2482.507568	10.300020	2964.972462	201.696720
max	3618.732910	25.206011	3600.000000	359.997589

Для определения наиболее подходящих графиков для анализа данных, построим парные диаграммы:

```
sns.pairplot(data_wind[data_wind.columns[1:]])
```



3.3 Выбор признаков, подходящих для построения моделей. Кодирование категориальных признаков. Масштабирование данных. Формирование вспомогательных признаков, улучшающих качество моделей.

Согласно типам данных, в нашем датасете присутствует колонка с типом данных object. Переведем ее в тип дата-время.

data_wind.dtypes

```
Date/Time                        object
LV ActivePower (kW)             float64
Wind Speed (m/s)                float64
Theoretical_Power_Curve (KWh)   float64
Wind Direction (°)              float64
dtype: object
```

pd.to_datetime(data_wind['Date/Time'])

```
0       2018-01-01 00:00:00
1       2018-01-01 00:10:00
2       2018-01-01 00:20:00
3       2018-01-01 00:30:00
4       2018-01-01 00:40:00
                ...
50525   2018-12-31 23:10:00
50526   2018-12-31 23:20:00
50527   2018-12-31 23:30:00
50528   2018-12-31 23:40:00
50529   2018-12-31 23:50:00
Name: Date/Time, Length: 50530, dtype: datetime64[ns]
```

Теперь наши данные в первой колонке имеют вид дата-время (timestamp).

3.4 Проведение корреляционного анализа данных. Формирование промежуточных выводов о возможности построения моделей машинного обучения.

Построим таблицы корреляции различными методами: Методом Пирсона, Кендала, Спирмана.

Классическое построение корреляционной модели будет производиться командой

data_wind.corr()

	LV ActivePower (kW)	Wind Speed (m/s)	Theoretical_Power_Curve (KWh)
LV ActivePower (kW)	1.000000	0.912774	0.949918
Wind Speed (m/s)	0.912774	1.000000	0.944209
Theoretical_Power_Curve (KWh)	0.949918	0.944209	1.000000

Корреляционный анализ методом Пирсона ищет взаимосвязь между двумя переменными в метрической шкале выбранной выборки. Коэффициент корреляции рассчитывается в пределах от минус единицы до плюс единицы. Таким образом Корреляция Пирсона рассчитывает линейную зависимость между величинами. Если связь криволинейная, то полученные значения будут некорректны. Формула, по которой рассчитывается коэффициенты корреляции представляет собой:

$$r_{xy} = \frac{\sum_{i=1}^{m}(x_i - \overline{x})(y_i - \overline{y})}{\sqrt{\sum_{i=1}^{m}(x_i - \overline{x})^2 \sum_{i=1}^{m}(y_i - \overline{y})^2}} = \frac{cov(x,y)}{\sqrt{s_x^2 s_y^2}},$$

Где $\overline{x}$ и $\overline{y}$ выборочные средние x^m и y^m, s_x^2, s_y^2 – выборочные дисперсии.

Или можно обозначить:

$\overline{X} = \frac{1}{n}\sum_{t=1}^{n} X_t$, $\overline{Y} = \frac{1}{n}\sum_{t=1}^{n} Y_t$ — среднее значение выборок.

Полученная корреляция data_wind.corr(method='pearson'), имеет такие же значения, как и при data_wind.corr()

Корреляционный анализ методом Кендала выполняется на основании оценок силы связей на основе рангов. Для вычисления коэффициентов используются не числовые значения, а ранги.

Выполним построение модели корреляционного анализа. Формула корреляционного анализа методом Кендала имеет вид:

$\tau = 1 - \frac{4}{n(n-1)}R$, где $R = \sum_{i=1}^{n-1}\sum_{j=i+1}^{n}\left[\left[x_i < x_j\right] \neq \left[y_i < y_j\right]\right]$.

Количество инверсий, в порядке возрастания x_i зависят от величин y_i .

Коэффициент, как и в случае Пирсона принимает значения от минус еденицы до еденицы. Если Т будет равно 1, значит это строгая линейная зависимость, -1 обратная линейная зависимость.

Построим корреляционный анализ командой

data_wind.corr(method='kendall')

	LV ActivePower (kW)	Wind Speed (m/s)	Theoretical_Power_Curve (KWh)
LV ActivePower (kW)	1.000000	0.862788	0.874933
Wind Speed (m/s)	0.862788	1.000000	0.982237
Theoretical_Power_Curve (KWh)	0.874933	0.982237	1.000000

Корреляционный анализ методом Спирмена это тоже линейная зависимость значений, она также является ранговой, как и корреляция Кендела. Для оценки используются не числовые значения, а значения рангов. Коэффициент корреляции Спирмена считается по формуле:

$$\rho = 1 - \frac{6}{n(n-1)(n+1)} \sum_{i=1}^{n} (R_i - S_i)^2,$$

Где R_i ранг наблюдения для x_i, а S_i ранг наблюдения для y_i.

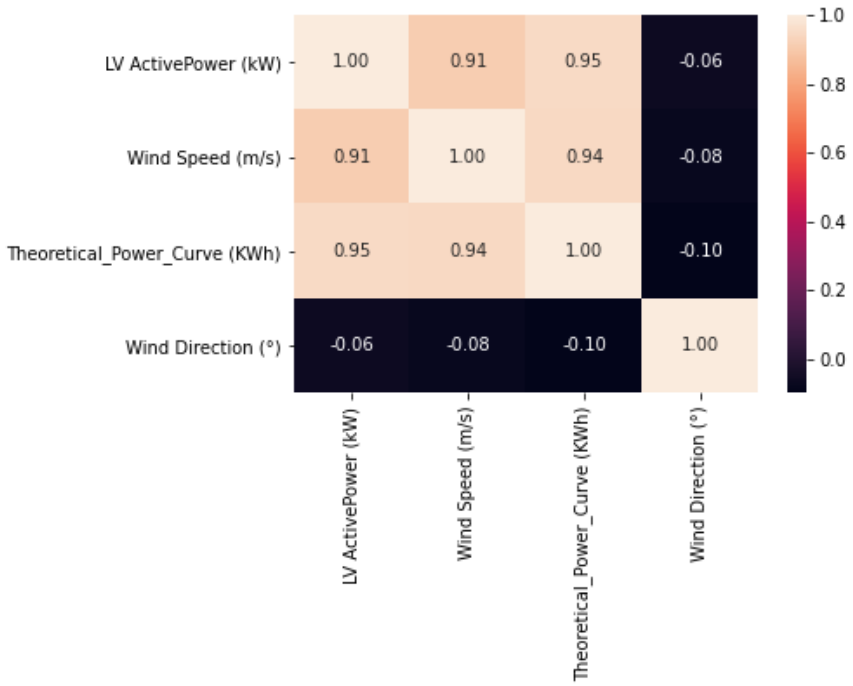
Построим модель корреляционного анализа методом Спирмена:

data_wind.corr(method='spearman')

	LV ActivePower (kW)	Wind Speed (m/s)	Theoretical_Power_Curve (KWh)
LV ActivePower (kW)	1.000000	0.932875	0.933979
Wind Speed (m/s)	0.932875	1.000000	0.997565
Theoretical_Power_Curve (KWh)	0.933979	0.997565	1.000000

Для более понятного отображения данных, построим корреляционную таблицу с помощью средств визуализации:

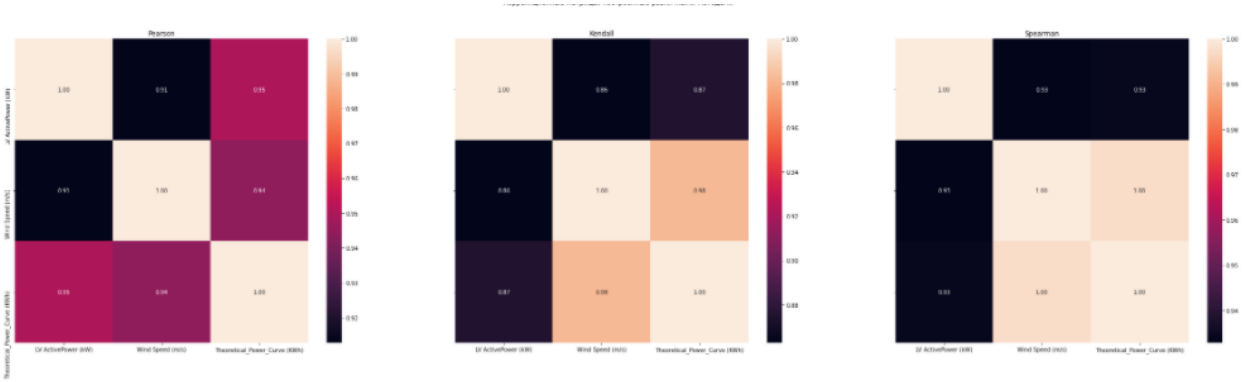
sns.heatmap(data_wind.corr(), annot=True, fmt='.2f')



На основании этой диаграммы, построим для каждого метода такие графики:

fig, ax = plt.subplots(1, 3, sharex='col', sharey='row', figsize=(40,10))
sns.heatmap(data_wind.corr(method='pearson'), ax=ax[0], annot=True, fmt='.2f')
sns.heatmap(data_wind.corr(method='kendall'), ax=ax[1], annot=True, fmt='.2f')
sns.heatmap(data_wind.corr(method='spearman'), ax=ax[2], annot=True, fmt='.2f')

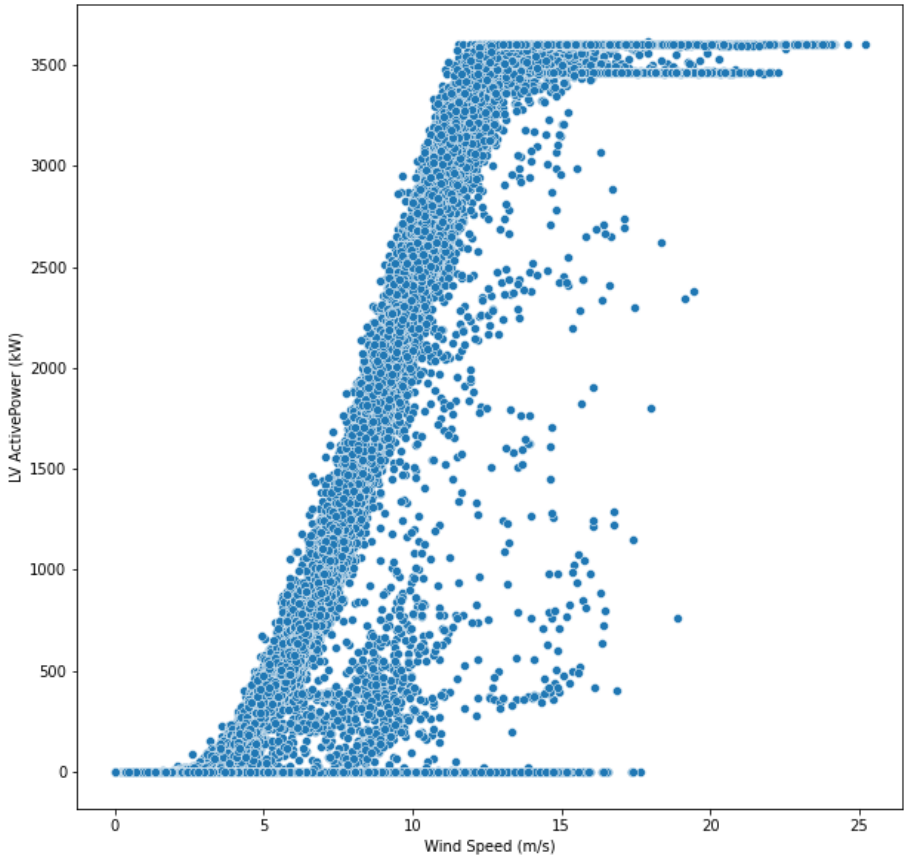
fig.suptitle('Корреляционные матрицы, построенные различными методами')
ax[0].title.set_text('Pearson')
ax[1].title.set_text('Kendall')
ax[2].title.set_text('Spearman')



На графиках видны наиболее коррелирующие значения.
Согласно отображаемой диаграмме корреляционного анализа, можно сделать вывод, что наибольшая корреляция проходит между значениями скорости ветра (Wind Speed) и Теоретической мощностью (Theoretical power), а также между Теоретической мощностью (Theoretical power) и Текущей скоростью (Active power), Наименьшая корреляция присутствует между направлением ветра и всеми остальными параметрами.

Таким образом, можно исключить переменную Wind direction (Направление ветра) из дальнейшего анализа.
data_wind = data_wind.drop(['Wind Direction (°)'], axis=1)
После удаления колонки, рассмотрим распределения переменных с наибольшим коэффициентом корреляции с целевой переменной.

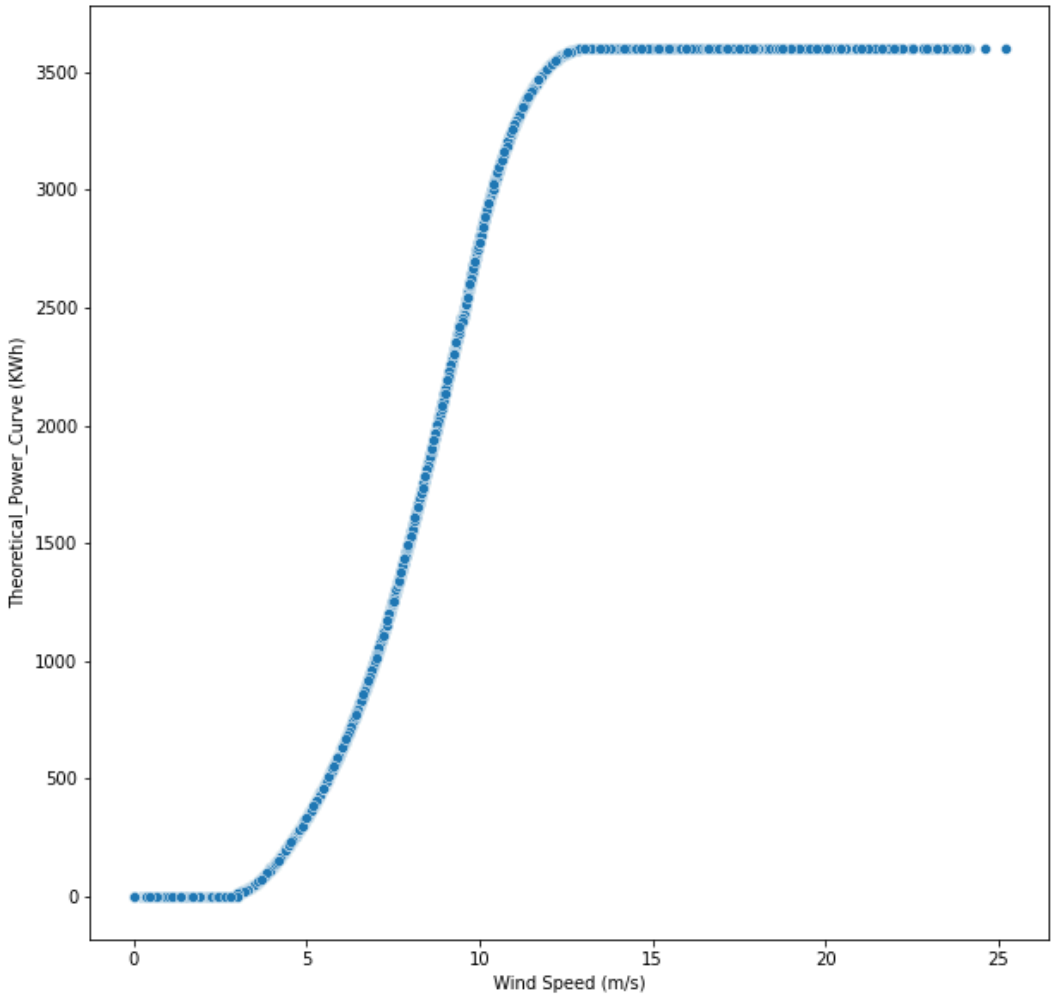
Скорость ветра и Активная мощность
fig, ax = plt.subplots(figsize=(10,10))
sns.scatterplot(ax=ax, x='Wind Speed (m/s)', y='LV ActivePower (kW)', data=data_wind)



Скорость ветра и Теоретическая мощность

fig, ax = plt.subplots(figsize=(10,10))

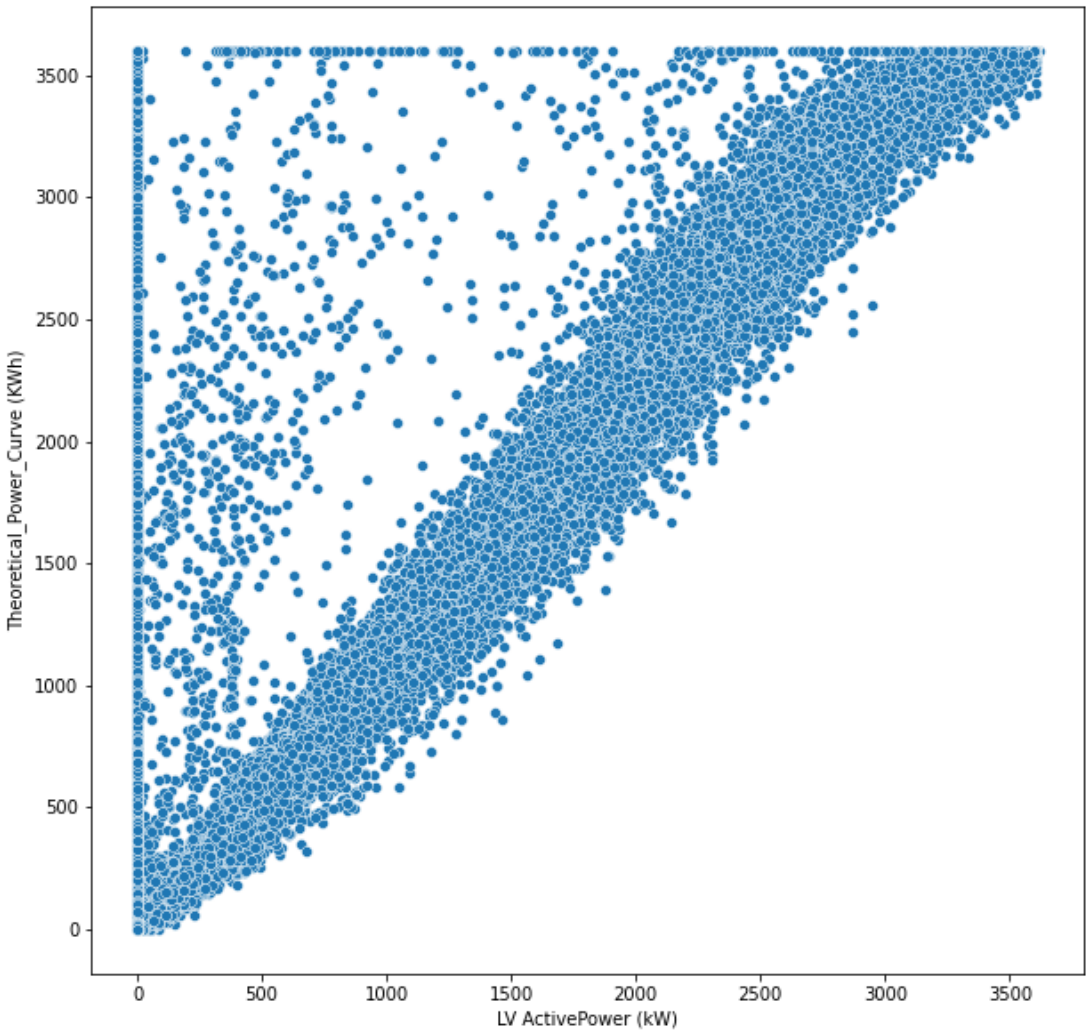
sns.scatterplot(ax=ax, x='Wind Speed (m/s)', y='Theoretical_Power_Curve (KWh)', data=data_wind)



Активная мощность и Теоретическая мощность

fig, ax = plt.subplots(figsize=(10,10)

sns.scatterplot(ax=ax, x='LV ActivePower (kW)', y='Theoretical_Power_Curve (KWh)', data=data_wind)

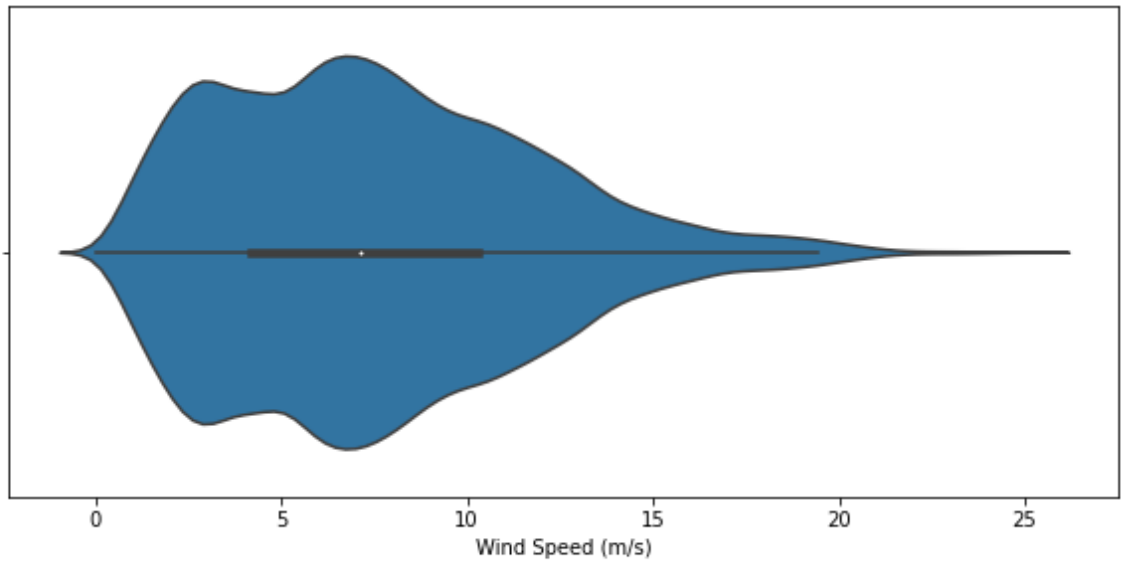


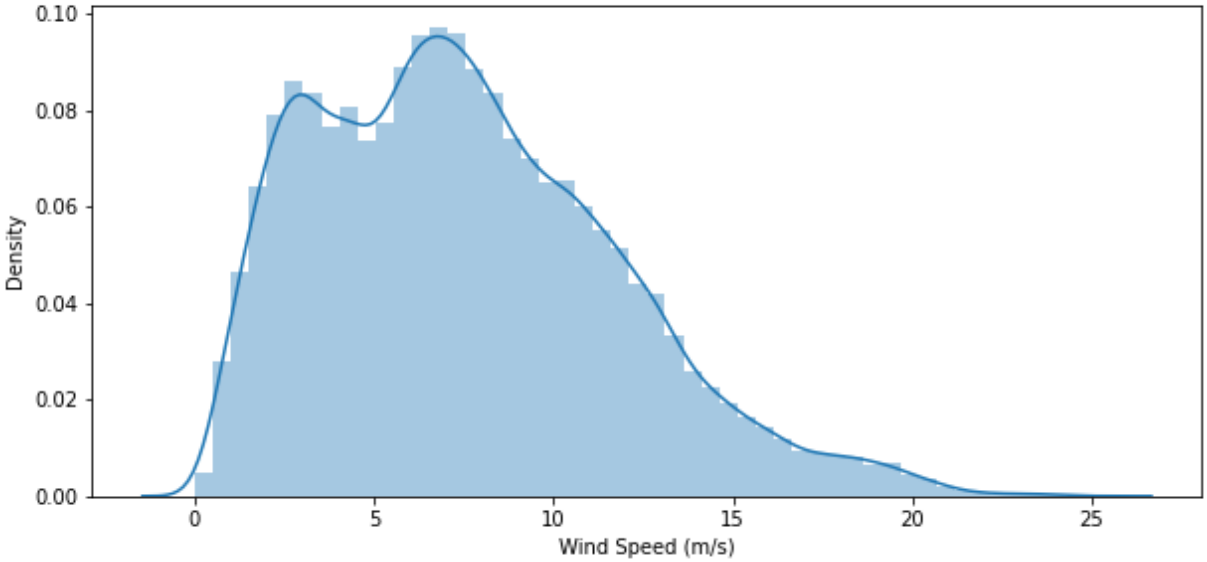
Нам интересны графики зависимости Скорости ветра от Активной мощности и Скорости ветра и Теоретической мощности

На графиках видно, что после увеличения скорости ветра с 5 м/с до ~12,5 м/с Активная мощность держится четким трендом. Эти два графика схожи по своему виду, однако на первом графике видны разбросы значений. Вероятно на выработку мощности влияет порывы ветра либо погрешности расчётов Теоретической мощности. На такие показатели также могут влиять климатические циклы.

Чтобы посмотреть значения скорости ветра в нашем наборе данных, воспользуемся

```
fig, ax = plt.subplots(2, 1, figsize=(10,10))
sns.violinplot(ax=ax[0], x=data_wind['Wind Speed (m/s)'])
sns.distplot(data_wind['Wind Speed (m/s)'], ax=ax[1])
```





Видно что больше всего значений скорости ветра присутствует в диапазоне от 4 до 10 м/с

Вывод: В данной работе продемонстрирована работа с данными, предобработка и первичный анализ для дальнейшей работы с набором данных. В следствии действий, были очищены, удалены, заменены некорректные данные, что в свою очередь приведет к наиболее корректным и приближенным значениям.

Список использованных источников

1. Python для анализа данных: обработка данных с помощью Pandas, NumPy и Python. Уэс МакКинни. 2-е издание от 24.10.2017 г.
2. Data Science. Наука о данных с нуля. Джоэл Грас. Год издания: 2017
3. Python для сложных задач: наука о данных и машинное обучение. Плас Дж. Вандер. Год издания: 2018
4. Введение в анализ данных с помощью Pandas URL: https://habr.com/ru/post/196980/ (Дата обращения: 20.10.2020)
5. Репозиторий курса "Методы машинного обучения", магистратура, 2 семестр. URL: https://github.com/ugapanyuk/ml_course_2020/wiki/COURSE_MMO (Дата обращения: 05.11.2020)