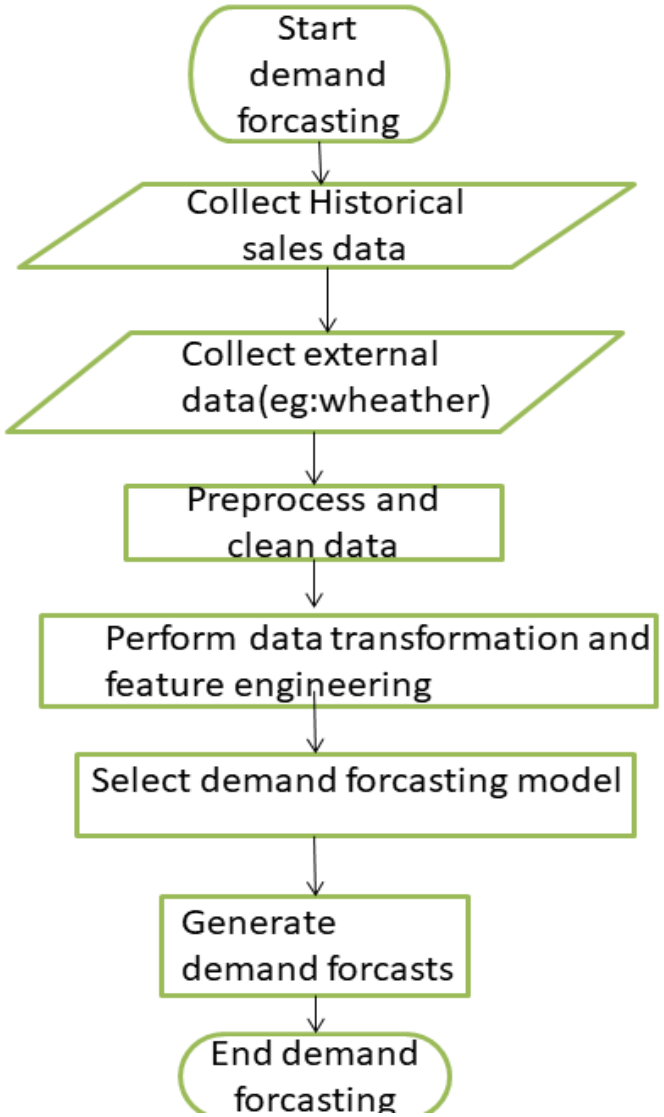
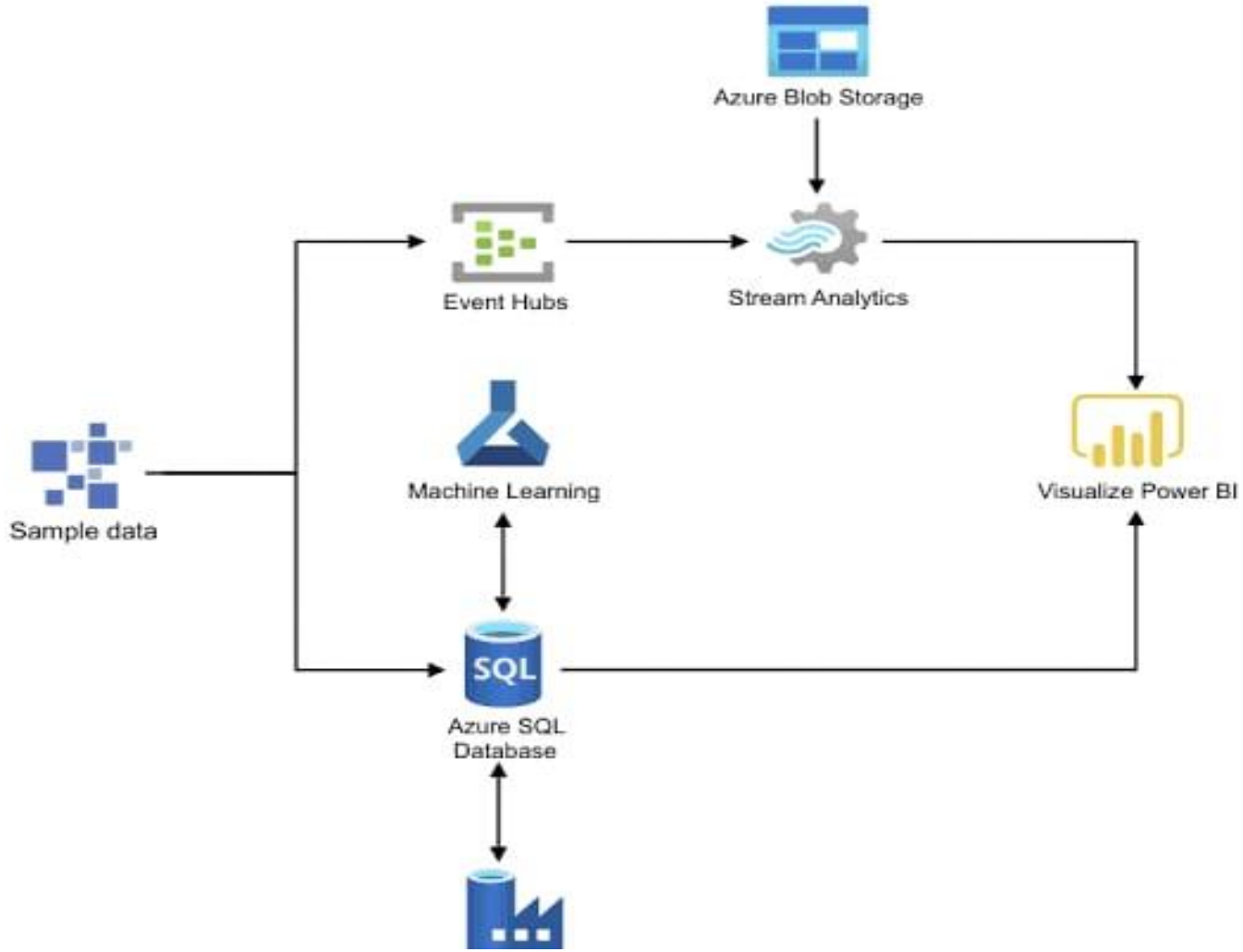
Demand Forecasting for E - Commerce

Flow Chart


Start demand forcasting
Collect Historical sales data
Collect external data(eg:wheather)
Preprocess and clean data
Perform data transformation and feature engineering
Select demand forcasting model
Generate demand forcasts
End demand forcasting

Architecture Diagram



Program Code

In [3]:
```
import numpy as np
import pandas as pd
from easypreprocessing import EasyPreProcessing
import seaborn as sns
import matplotlib
import matplotlib.pyplot as plt
import math
import scipy.stats as stats
```

In [4]:
```
prep = EasyPreProcessing('data.csv')
```

Initialization Parameters

1. output - Set output variable/dependent variable
2. dates.features - Set datetime field names (optional)

For example:

1. output = 'column_name'
2. dates.features = ['date_field_1','date_field_2']

In [5]:
```
prep.info
```

General Template

```
from easypreprocessing import EasyPreProcessing
prep = EasyPreProcessing('filename_here.csv')
prep.df
prep.output = 'output_variable_here'
prep.remove_blank()
prep.missing_values
prep.categorical.impute()
prep.numerical.impute()
prep.categorical.encode()
prep.correlation()
prep.standardize()
X_train, X_test, y_train, y_test = prep.split()
```

Categorical Preprocessing

categorical.fields	Display all categorical field names
categorical.unique	Display unique/distinct categorical values

In [8]:
```
prep.missing_values
```

Out[8]:
```
record_ID          0
week               0
store_id           0
sku_id             0
total_price        1
base_price         0
is_featured_sku    0
is_display_sku     0
units_sold         0
dtype: int64
```

In [9]:
```
prep.numerical.impute()
```

```
Numerical features imputated successfully.
```

In [10]:
```
prep.df
```

Out[10]:

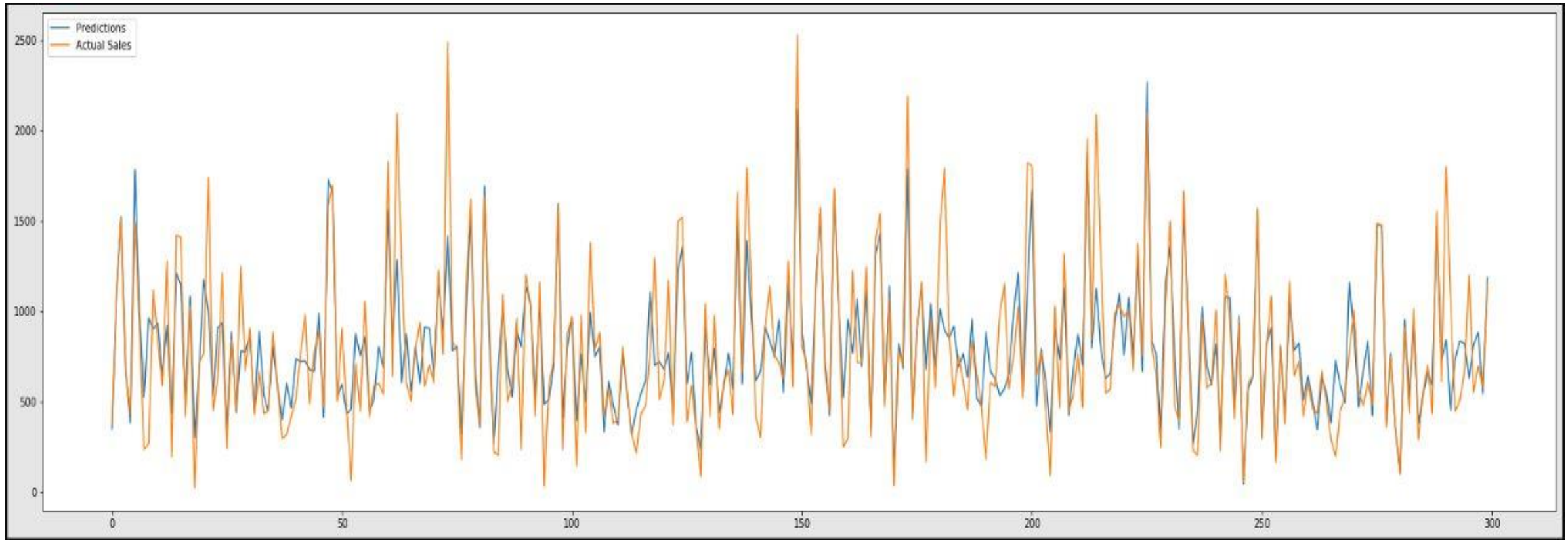
	record_ID	week	store_id	sku_id	total_price	base_price	is_featured_sku	is_display_sku	units_sold
0	1.0	17/01/11	8091.0	216418.0	99.0375	111.8625	0.0	0.0	20.0
1	2.0	17/01/11	8091.0	216419.0	99.0375	99.0375	0.0	0.0	28.0
2	3.0	17/01/11	8091.0	216425.0	133.9500	133.9500	0.0	0.0	19.0
3	4.0	17/01/11	8091.0	216233.0	133.9500	133.9500	0.0	0.0	44.0
4	5.0	17/01/11	8091.0	217390.0	141.0750	141.0750	0.0	0.0	52.0
...	...	...	...	...	...	...	...	...	...
150145	212638.0	09/07/13	9984.0	223245.0	235.8375	235.8375	0.0	0.0	38.0
150146	212639.0	09/07/13	9984.0	223153.0	235.8375	235.8375	0.0	0.0	30.0
150147	212642.0	09/07/13	9984.0	245338.0	357.6750	483.7875	1.0	1.0	31.0
150148	212643.0	09/07/13	9984.0	547934.0	141.7875	191.6625	0.0	1.0	12.0
150149	212644.0	09/07/13	9984.0	679023.0	234.4125	234.4125	0.0	0.0	15.0

150150 rows × 9 columns

In [11]:
```
prep.dataset['key'] = prep.df['week'].astype(str) + '_' + prep.df['store_id'].astype(str)
```

In [12]:
```
prep.dataset = prep.df.drop(['record_ID', 'week', 'store_id', 'sku_id', 'total_price', 'base_price', 'is_featured_sku', 'is_disp
```

Output


Predictions
Actual Sales
2500
2000
1500
1000
500
0
0
50
100
150
200
250
300