```python
#ricopue's notebook's code snippet
from sklearn.model_selection import StratifiedKFold
X_new=df_new.loc[train_index][feats]
y=df_new.loc[train_index]['preds']

params_lgb = {'learning_rate': 0.06,'objective': 'multiclass','boosting': 'gbdt','n_jobs': -1,'verbosity': -1, 'num_classes':7}

model_list=[]

gkf = StratifiedKFold(11)
for fold, (train_idx, valid_idx) in enumerate(gkf.split(X_new,y)):

    tr_dataset = lgb.Dataset(X_new.iloc[train_idx],y.iloc[train_idx],feature_name = feats)
    vl_dataset = lgb.Dataset(X_new.iloc[valid_idx],y.iloc[valid_idx],feature_name = feats)

    model = lgb.train(params = params_lgb,
                train_set = tr_dataset,
                valid_sets =  vl_dataset,
                num_boost_round = 5000,
                callbacks=[ lgb.early_stopping(stopping_rounds=300, verbose=False), lgb.log_evaluation(period=200)])

    model_list.append(model)
```

```
[200]	valid_0's multi_logloss: 0.000151038
[400]	valid_0's multi_logloss: 0.000151038
[200]	valid_0's multi_logloss: 0.000151576
[400]	valid_0's multi_logloss: 0.000151576
[200]	valid_0's multi_logloss: 0.00015433
[400]	valid_0's multi_logloss: 0.00015433
[200]	valid_0's multi_logloss: 0.000155486
[400]	valid_0's multi_logloss: 0.000155486
[200]	valid_0's multi_logloss: 0.000153559
[400]	valid_0's multi_logloss: 0.000153559
[200]	valid_0's multi_logloss: 0.000155711
[400]	valid_0's multi_logloss: 0.000155711
[200]	valid_0's multi_logloss: 8.58365e-06
```

```
[3000]	valid_0's multi_logloss: 8.20875e-06
[3200]	valid_0's multi_logloss: 8.20327e-06
[3400]	valid_0's multi_logloss: 8.19834e-06
[3600]	valid_0's multi_logloss: 8.19389e-06
[3800]	valid_0's multi_logloss: 8.18985e-06
[4000]	valid_0's multi_logloss: 8.18617e-06
[4200]	valid_0's multi_logloss: 8.1828e-06
[4400]	valid_0's multi_logloss: 8.1797e-06
[4600]	valid_0's multi_logloss: 8.17684e-06
[4800]	valid_0's multi_logloss: 8.1742e-06
[5000]	valid_0's multi_logloss: 8.17175e-06
[200]	valid_0's multi_logloss: 0.000220707
[400]	valid_0's multi_logloss: 0.000220707
[200]	valid_0's multi_logloss: 0.000156024
[400]	valid_0's multi_logloss: 0.000156024
[200]	valid_0's multi_logloss: 0.0120099
[400]	valid_0's multi_logloss: 0.0120099
[200]	valid_0's multi_logloss: 0.0238626
```

+ Code + Markdown

[121]:

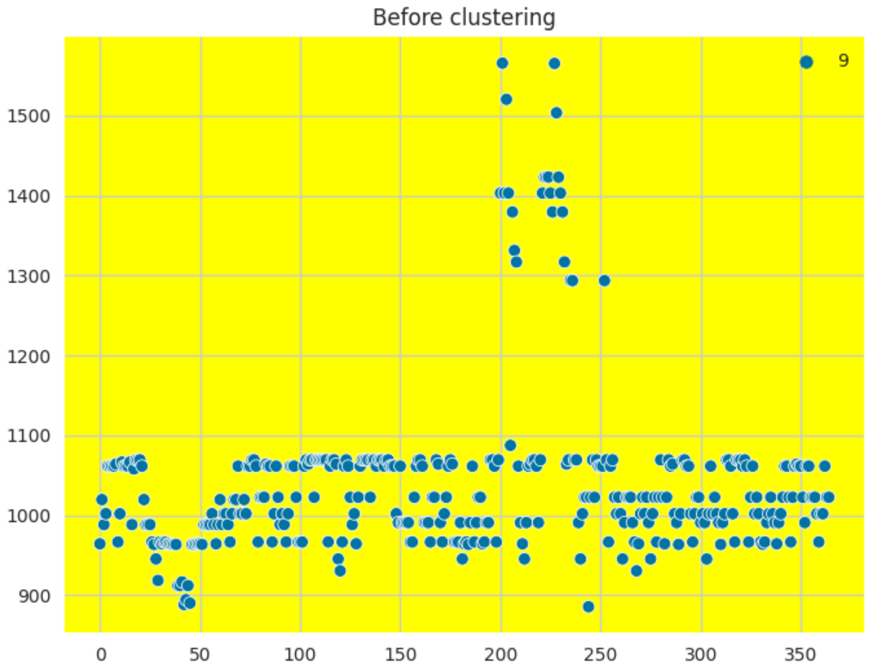
```python
lgb_preds=0
for model in model_list:
    lgb_preds+=model.predict(df_new[feats])
```

[125]:

```python
labels=np.argmax(lgb_preds,axis=1)
```

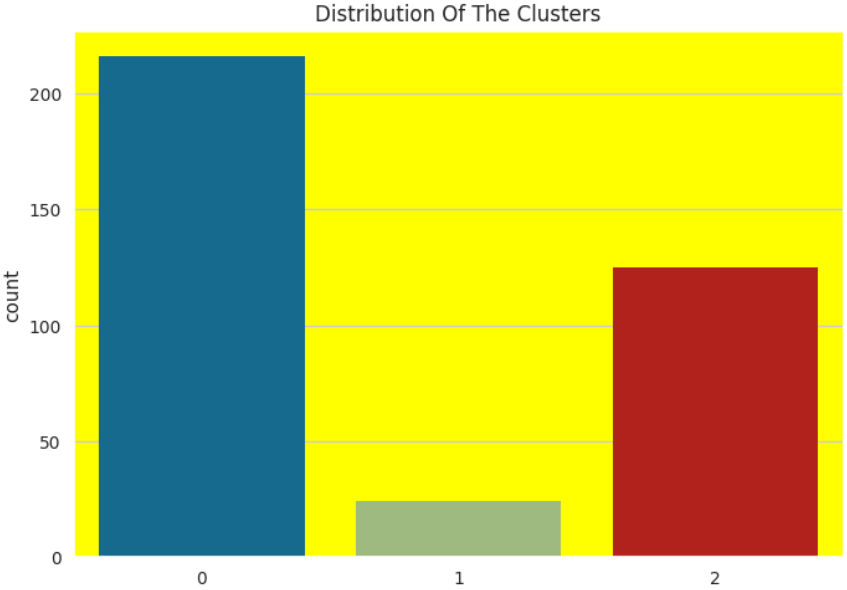
[127]:

```python
fig = plt.figure(figsize=(8,6))
ax = plt.subplot(label="bla")
sns.scatterplot(df[feats], marker='o');
ax.set_title("Before clustering");
```



[132]:

```
pl = sns.countplot(x=np.argmax(lgb_preds,axis=1))
pl.set_title("Distribution Of The Clusters")
plt.show()
```



[134]:

```
import tensorflow as tf
from keras import Model
from keras.layers import Input, Dense, Dropout
from keras.layers import LSTM
```

+ Code + Markdown

[136]:

```
df['Ngay'] = pd.to_datetime(df['Ngay'], format='%d/%m/%Y')
df.sort_values(by='Ngay', ascending=True, inplace=True)
df.reset_index(drop=True, inplace=True)
```

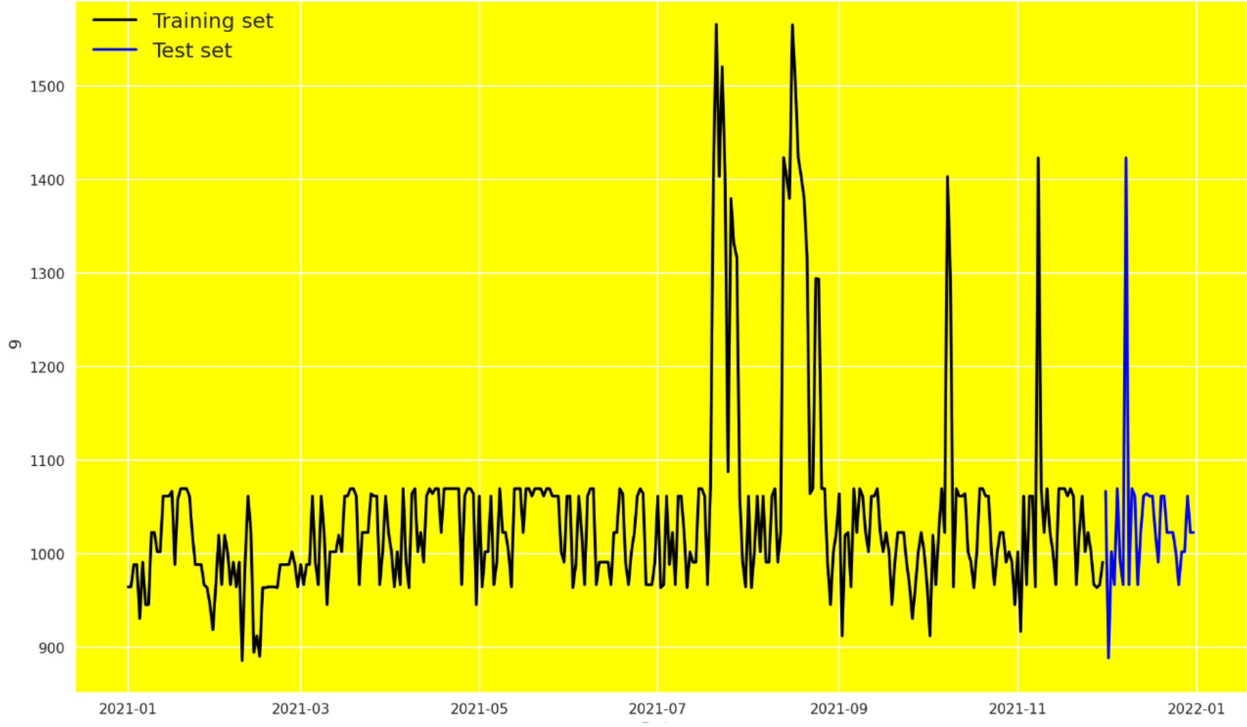
[137]:

```
test_size = df[df['Ngay'].dt.month==12].shape[0]
test_size
```

[137… 31

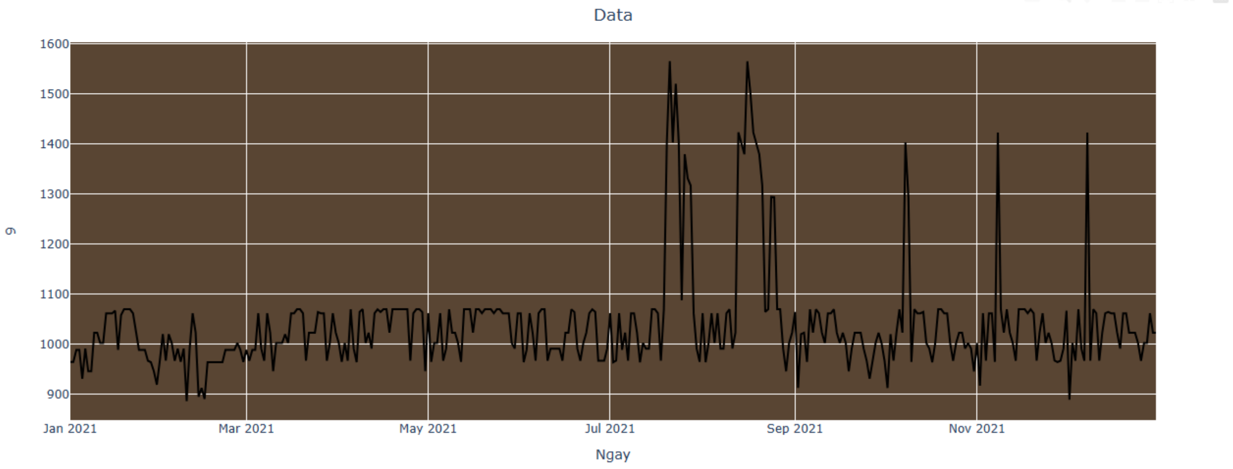
[138]:

```
plt.figure(figsize=(15, 9), dpi=150)
plt.rcParams['axes.facecolor'] = 'yellow'
plt.rc('axes',edgecolor='white')
plt.plot(df['Ngay'][:-test_size], df['9'][:-test_size], color='black', lw=2)
plt.plot(df['Ngay'][-test_size:], df['9'][-test_size:], color='blue', lw=2)
plt.title('9', fontsize=15)
plt.xlabel('Date', fontsize=12)
plt.ylabel('9', fontsize=12)
plt.legend(['Training set', 'Test set'], loc='upper left', prop={'size': 15})
plt.grid(color='white')
plt.show()
```



:

```
import plotly.express as px
fig = px.line(y=df['9'], x=df['Ngay'])
fig.update_traces(line_color='black')
fig.update_layout(xaxis_title="Ngay",
                  yaxis_title="9",
                  title={'text': "Data", 'y':0.95, 'x':0.5, 'xanchor':'center', 'yanchor':'top'},
                  plot_bgcolor='rgba(47,23,0,0.8)')
```



[141]:

```
from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler()
scaler.fit(df['9'].values.reshape(-1,1))
```

[141…

```
▾ MinMaxScaler
MinMaxScaler()
```

+ Code + Markdown

[142]:

```
window_size = 15
```

[144]:

```
train_data = df['9'][:-test_size]
train_data = scaler.transform(train_data.values.reshape(-1,1))
```

[145]:

```
X_train = []
y_train = []

for i in range(window_size, len(train_data)):
    X_train.append(train_data[i-window_size:i, 0])
    y_train.append(train_data[i, 0])
```

147]:

```
test_data = df['9'][-test_size-window_size:]
test_data = scaler.transform(test_data.values.reshape(-1,1))
```

148]:

```
X_test = []
y_test = []

for i in range(window_size, len(test_data)):
    X_test.append(test_data[i-window_size:i, 0])
    y_test.append(test_data[i, 0])
```

+ Code + Markdown

151]:

```
X_train = np.array(X_train)
X_test  = np.array(X_test)
y_train = np.array(y_train)
y_test  = np.array(y_test)
```

152]:

```
X_train = np.reshape(X_train, (X_train.shape[0], X_train.shape[1], 1))
X_test  = np.reshape(X_test, (X_test.shape[0], X_test.shape[1], 1))
y_train = np.reshape(y_train, (-1,1))
y_test  = np.reshape(y_test, (-1,1))
```

+ Code + Markdown

[154]:

```
model = define_model()
history = model.fit(X_train, y_train, epochs=100, batch_size=16, validation_split=0.1, verbose=1)
```

```
Model: "functional_5"

┏━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━┓
┃ Layer (type)                    ┃ Output Shape           ┃       Param # ┃
┡━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━┩
│ input_layer_2 (InputLayer)      │ (None, 15, 1)          │             0 │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ lstm_2 (LSTM)                   │ (None, 64)             │        16,896 │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ dense_4 (Dense)                 │ (None, 32)             │         2,080 │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ dense_5 (Dense)                 │ (None, 1)              │            33 │
└─────────────────────────────────┴────────────────────────┴───────────────┘

 Total params: 19,009 (74.25 KB)

 Trainable params: 19,009 (74.25 KB)

 Non-trainable params: 0 (0.00 B)

Epoch 1/100
18/18 ━━━━━━━━━━━━━━━━━━━━ 3s 24ms/step - loss: 0.1202 - val_loss: 0.0482
Epoch 2/100
18/18 ━━━━━━━━━━━━━━━━━━━━ 0s 9ms/step - loss: 0.0307 - val_loss: 0.0187
Epoch 3/100
18/18 ━━━━━━━━━━━━━━━━━━━━ 0s 8ms/step - loss: 0.0225 - val_loss: 0.0155
Epoch 4/100
18/18 ━━━━━━━━━━━━━━━━━━━━ 0s 10ms/step - loss: 0.0268 - val_loss: 0.0164
Epoch 5/100
18/18 ━━━━━━━━━━━━━━━━━━━━ 0s 9ms/step - loss: 0.0243 - val_loss: 0.0158
Epoch 6/100
18/18 ━━━━━━━━━━━━━━━━━━━━ 0s 9ms/step - loss: 0.0221 - val_loss: 0.0164
Epoch 7/100
18/18 ━━━━━━━━━━━━━━━━━━━━ 0s 8ms/step - loss: 0.0265 - val_loss: 0.0156
Epoch 8/100
18/18 ━━━━━━━━━━━━━━━━━━━━ 0s 9ms/step - loss: 0.0300 - val_loss: 0.0156
Epoch 9/100
Epoch 93/100
18/18 ━━━━━━━━━━━━━━━━━━━━ 0s 9ms/step - loss: 0.0099 - val_loss: 0.0201
Epoch 94/100
18/18 ━━━━━━━━━━━━━━━━━━━━ 0s 8ms/step - loss: 0.0132 - val_loss: 0.0189
Epoch 95/100
18/18 ━━━━━━━━━━━━━━━━━━━━ 0s 9ms/step - loss: 0.0107 - val_loss: 0.0190
Epoch 96/100
18/18 ━━━━━━━━━━━━━━━━━━━━ 0s 8ms/step - loss: 0.0098 - val_loss: 0.0204
Epoch 97/100
18/18 ━━━━━━━━━━━━━━━━━━━━ 0s 9ms/step - loss: 0.0104 - val_loss: 0.0200
Epoch 98/100
18/18 ━━━━━━━━━━━━━━━━━━━━ 0s 9ms/step - loss: 0.0126 - val_loss: 0.0203
Epoch 99/100
18/18 ━━━━━━━━━━━━━━━━━━━━ 0s 8ms/step - loss: 0.0095 - val_loss: 0.0185
Epoch 100/100
18/18 ━━━━━━━━━━━━━━━━━━━━ 0s 9ms/step - loss: 0.0112 - val_loss: 0.0214
```

+ Code + Markdown

[155]:

```python
result = model.evaluate(X_test, y_test)
y_pred = model.predict(X_test)
```

```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 251ms/step - loss: 0.0231
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 189ms/step
```

[156]:

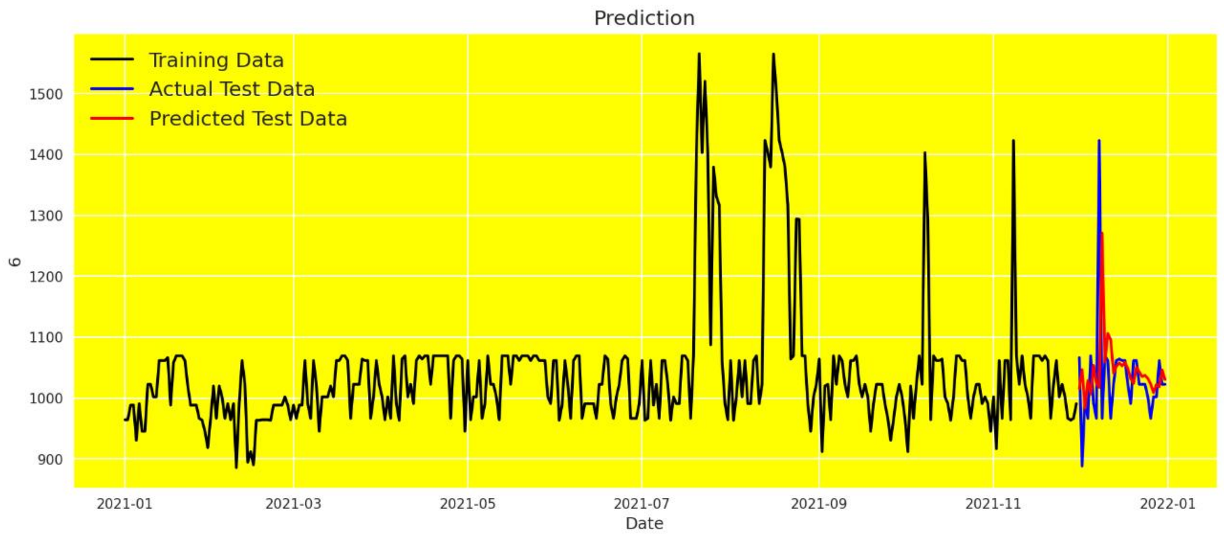
```python
from sklearn.metrics import mean_absolute_percentage_error,accuracy_score,r2_score
MAPE = mean_absolute_percentage_error(y_test, y_pred)
Accuracy = 1-MAPE
```

[157]:

```python
print("Test Loss:", result)
print("Test MAPE:", MAPE)
print("Test Accuracy:", Accuracy)
```

```
Test Loss: 0.02308393269777298
Test MAPE: 2.2222021315315548
Test Accuracy: -1.2222021315315548
```

```python
plt.figure(figsize=(15, 6), dpi=150)
plt.rcParams['axes.facecolor'] = 'yellow'
plt.rc('axes',edgecolor='white')
plt.plot(df['Ngay'].iloc[:-test_size], scaler.inverse_transform(train_data), color='black', lw=2)
plt.plot(df['Ngay'].iloc[-test_size:], y_test_true, color='blue', lw=2)
plt.plot(df['Ngay'].iloc[-test_size:], y_test_pred, color='red', lw=2)
plt.title('Prediction', fontsize=15)
plt.xlabel('Date', fontsize=12)
plt.ylabel('6', fontsize=12)
plt.legend(['Training Data', 'Actual Test Data', 'Predicted Test Data'], loc='upper left', prop={'size': 15})
plt.grid(color='white')
plt.show()
```


Prediction
Training Data
Actual Test Data
Predicted Test Data
Date