

```
In [12]: import warnings
warnings.filterwarnings('ignore')
warnings.warn("deprecated", DeprecationWarning)

import pandas as pd
import numpy as np
import seaborn as sns
import scipy.stats as stats

import pickle
```

```
In [2]: def final(test_df):
        iterativeImputr = pickle.load(open('iterative_imputer.pickle', 'rb'))
        test_df = iterativeImputr.transform(test_df)
        robustscaler = pickle.load(open('robust_scaler.pickle', 'rb'))
        test_df = robustscaler.transform(test_df)
        loaded_model = pickle.load(open('randomforest_model.pickle', 'rb'))
        prediction = loaded_model.predict(test_df)

        return prediction
```

```
In [27]: test = pd.read_csv('test.csv')
test = test.drop(['sku', 'went_on_backorder'],axis=1)
test_cpy = test.copy()

categorical_features = []
for col in test.columns:
    if (test.dtypes[col] == 'object'):
        categorical_features.append(col)

for feature in categorical_features:
    if(feature != 'sku'):
        test[feature] = test[feature].map({"Yes" : 1, "No" : 0})

predict = final(test.iloc[[5]])
print(predict)

[0]
```

Following contain the link of the deployment video

<https://drive.google.com/file/d/1zj0YPc7nQqV0dEZw7h9Gxg9yd0DJiZGZ/view?usp=sharing>
(<https://drive.google.com/file/d/1zj0YPc7nQqV0dEZw7h9Gxg9yd0DJiZGZ/view?usp=sharing>)

Code for Deployment

```

In [ ]: #importing libraries
import warnings
warnings.filterwarnings('ignore')
warnings.warn("deprecated", DeprecationWarning)

import pandas as pd
import numpy as np
import seaborn as sns
import scipy.stats as stats

import pickle
from flask import Flask, request, render_template, send_file

app = Flask(__name__)

@app.route("/")
def home():
    return render_template('index.html')

@app.route('/predict', methods = ['GET','POST'])
def predict_backorder():
    try:
        # Read the uploaded csv file
        file = request.files['search_file']
        test_df = pd.read_csv(file)
        # Drop the dependent variables and keep a copy of the csv file for the
        final csv download
        test_df = test_df.drop(['went_on_backorder'],axis=1)
        test_df_cpy_with_sku = test_df.copy()
        test_df = test_df.drop(['sku'],axis=1)
        # Find out the categorical features for feature engineering
        categorical_features = []
        for col in test_df.columns:
            if (test_df.dtypes[col] == 'object'):
                categorical_features.append(col)
        # Preprocess the categorical features and numarical features
        for feature in categorical_features:
            if(feature != 'sku'):
                test_df[feature] = test_df[feature].map({"Yes" : 1, "No" : 0})

        test_df.perf_12_month_avg.replace({-99.0 : np.nan}, inplace = True)
        test_df.perf_6_month_avg.replace({-99.0 : np.nan}, inplace = True)
        test_df['lead_time'].fillna(test_df['lead_time'].mean(),inplace=True)

        iterativeImputr = pickle.load(open('iterative_imputer.pickle', 'rb'))
        test_df = iterativeImputr.transform(test_df)
        robusrscaler = pickle.load(open('robust_scaler.pickle', 'rb'))
        test_df = robusrscaler.transform(test_df)
        loaded_model = pickle.load(open('randomforest_model.pickle', 'rb'))
        prediction = loaded_model.predict(test_df)

        # Save the predicted dependent variable for final csv file creation
        final_df = pd.DataFrame()
        final_df['sku'] = test_df_cpy_with_sku['sku']
        final_df['went_on_backorder'] = prediction.tolist()
        final_df['went_on_backorder'] = final_df['went_on_backorder'].map({1 :

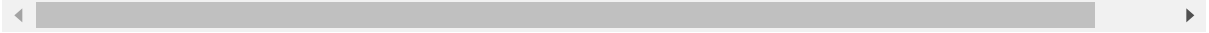
```

```
"Yes" , 0 : "No" })
    final_df.to_csv('final_csv.csv')

    return send_file('final_csv.csv', mimetype='csv', as_attachment=True,
attachment_filename='backorder_prediction_final.csv')

except Exception as e:
    return render_template('index.html',error=e)

if __name__ == '__main__':
    app.run(debug=True)
```

A horizontal scrollbar is located at the bottom of the code editor. It consists of a light gray track with a darker gray slider bar positioned towards the left side. Small arrowheads are visible at both ends of the track.