```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
import matplotlib.pyplot as plt
from matplotlib.widgets import TextBox

# Load the CSV file
df = pd.read_csv('Salary_dataset.csv')

# Selecting the feature and target variable
X = df[['YearsExperience']]
y = df['Salary']

# Splitting the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=0)

# Creating the Linear Regression model
model = LinearRegression()

# Training the model
model.fit(X_train, y_train)

# Predicting the salaries for the test set
y_pred = model.predict(X_test)

# Plotting with interactive TextBox
fig, ax = plt.subplots()
plt.subplots_adjust(bottom=0.2)  # Adjust the bottom
ax.scatter(X_train, y_train, color='red', label='Training Data')
ax.scatter(X_test, y_test, color='green', label='Test Data')
line, = ax.plot(X_train, model.predict(X_train), label='Regression Line')
estimated_point, = ax.plot([], [], 'ko', label='Estimated Data')  # 'ko' creates a black circle marker

ax.set_title('Salary vs Experience')
ax.set_xlabel('Years of Experience')
ax.set_ylabel('Salary')
ax.legend(loc='upper center', bbox_to_anchor=(0.5, -0.2), shadow=True, ncol=2,
fontsize='small')


# Textbox location and properties
text_box_ax = plt.axes([0.2, 0.05, 0.1, 0.05])  # x-position, y-position, width, height
```

```
text_box = TextBox(text_box_ax, 'Enter Years of Experience')

# Text annotation for displaying the prediction
prediction_text = ax.text(0.05, 0.95, '', transform=ax.transAxes)

def submit(text):
    years_exp = float(text)
    new_data = pd.DataFrame([[years_exp]], columns=['YearsExperience'])
    salary_pred = model.predict(new_data)[0]

     # Update the position of the estimated data point
    estimated_point.set_data([years_exp], [salary_pred])
    estimated_point.set_visible(True)  # Make the marker visible

    prediction_text.set_text(f'Estimated salary for {years_exp:.2f} years: ${salary_pred:.2f}')


    # Update regression line for new input
    new_X_train = pd.concat([X_train, new_data]).reset_index(drop=True)
    new_y_train = model.predict(new_X_train)
    line.set_xdata(new_X_train)
    line.set_ydata(new_y_train)

    # Redrawing the canvas
    fig.canvas.draw_idle()

# When the user submits input, the submit function is called
text_box.on_submit(submit)

plt.draw()

plt.show()
```