TITLE

Lab- 7: Embed developed Machine Learning Models to Web page.

Content

Lab -7

AIM:

Embed developed Machine Learning Models to Web page.

Description:

From the previous experiments we learn how to make an ml prediction model. Now, we will integrate our model to web page so that it can be used by user how knows only to interact with web page and not code.

In this we are using salary predictor based on the years of experience. When you type you expected experience year it will down show the expected salary a person of this much year of experience should get.

Code and Output:

```
1 from google.colab import drive
2 drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

Fig 1: Connection with google drive

	A	B	C
	YearsExperience	Age	Salary
	1.1	21	39343
	1.3	21.5	46205
	1.5	21.7	37731
	2	22	43525
	2.2	22.2	39891
	2.9	23	56642
	3	23	60150
	3.2	23.3	54445
	3.2	23.3	64445
	3.7	23.6	57189
	3.9	23.9	63218
	4	24	55794
	4	24	56957
	4.1	24	57081
	4.5	25	61111
	4.9	25	67938
	5.1	26	66029
	5.3	27	83088
	5.9	28	81363
	6	29	93940
	6.8	30	91738
	7.1	30	98273
	7.9	31	101302
	8.2	32	113812
	8.7	33	109431
	9	34	105582
	9.5	35	116969
	9.6	36	112635

```
import pandas as pd
import numpy as np

data = pd.read_csv('/content/drive/MyDrive/ML/Salary_Data.csv')

x = np.array(data['YearsExperience'].values)
print(x)
y = np.array(data['Salary'].values)
print(y)
```

```
[ 1.1  1.3  1.5  2.   2.2  2.9  3.   3.2  3.2  3.7  3.9  4.   4.   4.1
  4.5  4.9  5.1  5.3  5.9  6.   6.8  7.1  7.9  8.2  8.7  9.   9.5  9.6
 10.3 10.5]
[ 39343  46205  37731  43525  39891  56642  60150  54445  64445  57189
  63218  55794  56957  57081  61111  67938  66029  83088  81363  93940
  91738  98273 101302 113812 109431 105582 116969 112635 122391 121872]
```

Fig 2: Importing Data

```
[ ]
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(x, y, test_size=0.2, random_state=42)
```

```
from sklearn.linear_model import LinearRegression


reg = LinearRegression()
X_train = np.reshape(X_train, (-1, 1))

reg.fit(X_train, y_train)
```

```
LinearRegression()
```

Fig 3: Importing sklearn and performing linear regression

```
[ ]
import pickle
pickle.dump(reg, open('model.pkl','wb'))
model = pickle.load(open('model.pkl','rb'))

```

Fig 4: Creating model

```html
<!DOCTYPE html>
<html lang="en">
<head>

    <title>Salary Predictor</title>
</head>
<body>
    <h1>Salary Predictor</h1>
    <form action="{{ url_for('predict')}}" method="post">
     Enter Your Expected Year of Experience:    <input type="number" required = "required" name="salary">
        <button type="submit" class="btn">Predict</button>
    </form>


    <br><br>
    {{ prediction_text }}
</body>
</html>
```

```python
def home():
    return render_template('index.html')

@app.route('/predict', methods=['POST'])
def predict():
    # int_features = request.form['salary']
    # print( float(x) for x in request.form.values())
    final_features = float(request.form['salary'])

    prediction = model.predict([[final_features]])

    output = round(prediction[0], 2)
   # print(output)
    return render_template('index.html', prediction_text='Predicted Salary: {}'.format(output))

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

Fig 5: HTML and Flask code for integration

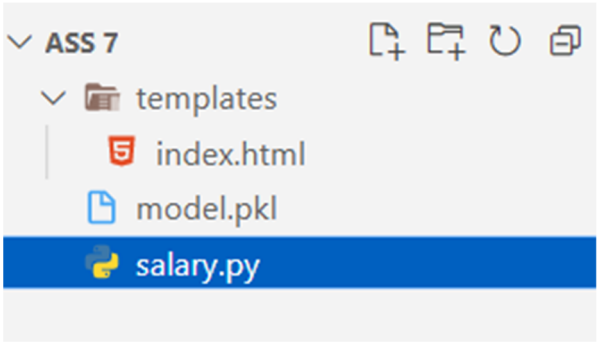

Fig 6: File Structure

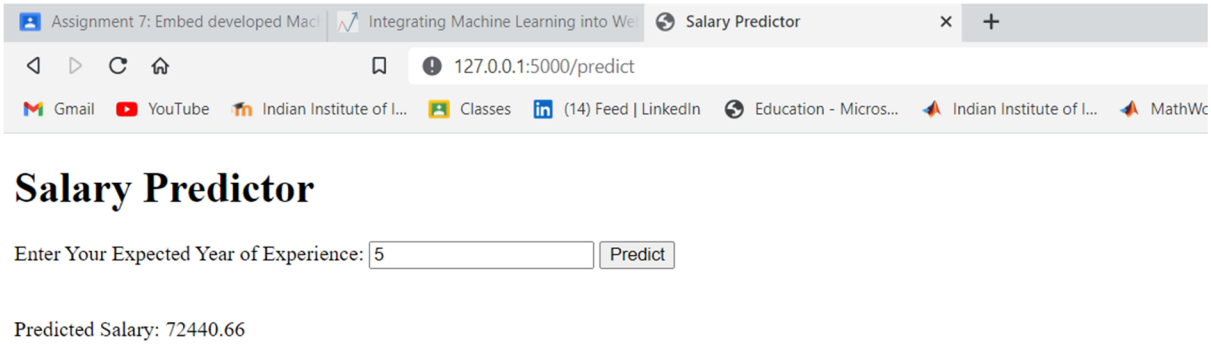

Fig 7: Output at localhost:5000

Conclusion:

The above model Predicts the salary of the user based on the year of experience however the prediction can be more precise if we train this on a large dataset.