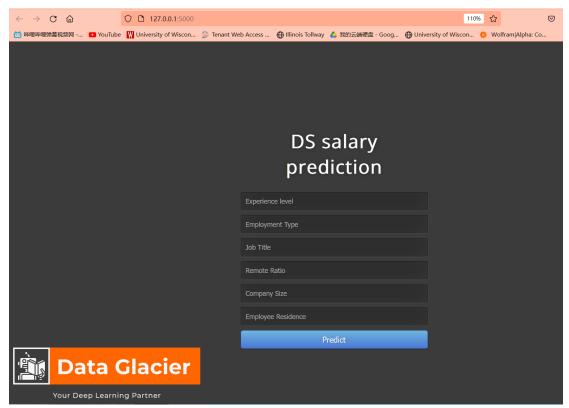
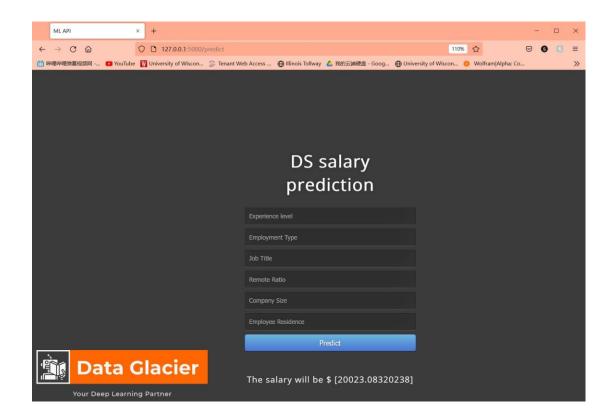
Name: Shunyi Huang

Submission Date: 09/04/2022







Logout

```
File Edit View Language
                                                                                                                                                                                                                                                                                                                                                           Python
    1 #!/usr/bin/env python
2 # coding: utf-8
           # In[1]:
           import pandas as pd
           import numpy as np
import matplotlib.pyplot as plt
           import seaborn as sns
           from sklearn.model_selection import train_test_split
           import pickle
           import requests
import json
           from statsmodels stats outliers influence import variance inflation factor
           from sklearn.preprocessing import LabelEncoder from sklearn.linear_model import LinearRegression
           from flask import Flask, request, jsonify, render_template from sklearn.metrics import mean_squared_error
23 # # EDA
24
 25 # In/27:
 28 def eda(df):
29 print(d
                          print(df.head(2),'\n')
 30
                            print(df.info())
                            print(df.describe(),'\n')
                          print(df.isna().sum())
print('Number of data point is: ', len(df))
 35 def box_plot(df, feature):
36 df.plot(y = feature, kind = 'box')
 38 def heat_map(df):
39 sns.heatmap(df.corr(),annot=True)
 40
  def check_vif(df, features):
                          check_vir(dr, reatures):
    df = df. drop(features, axis = 1)
    vif_data = pd. DataFrame()
    vif_data['feature'] = df. columns
 45
                           \begin{array}{lll} & vif\_data['VIF'] = [variance\_inflation\_factor(df.values,\ i)\ for\ i\ in\ range(len(df.columns))] \\ & print(vif\_data) \end{array} 
 50 # In/37:
 53 df = pd.DataFrame(pd.read_csv('ds_salaries.csv'))
54 #df.head()
        # In[4]:
 60 le = LabelEncoder()
60 le = LabelEncoder()
df'(salary_currency') = le.fit_transform(df['salary_currency'])
61 df'('employee_residence') = le.fit_transform(df['employee_residence'])
62 df['company_location'] = le.fit_transform(df['company_location'])
63 df['company_size'] = le.fit_transform(df['company_size'])
64 df['gompany_size'] = le.fit_transform(df['experience_level'])
65 df['job_title'] = le.fit_transform(df['sob_title'])
 69 # In[5]:
72 | df['employment_type'] = le.fit_transform(df['employment_type'])
73 | df.head()
74 |
 76 # In[6]:
 78
79 # eda(df)
           # heat man(df)
            # check_vif(df,['work_year','company_location','salary_currency'])
 83
84 ## Data splitting
          X = df.loc[:, df.columns != 'salary_in_usd']
X = X.drop(['Unnamed: 0', 'work_year', 'salary_currency', 'company_location'], axis = 1)
y = df['salary_in_usd']
91
92 # df = df.drop(['work_year','company_location','salary_currency','salary'], axis = 1)
93 # X = df.loc[:, df.columns != 'salary_in_usd']
94 # X = X.drop(['Unnamed: 0', 'experience_level', 'employment_type', 'employee_residence', 'remote_ratio', 'company_size'], axis = 1)
95 # y = df['salary_in_usd']
96 # Y tring Variation | Action | Actio
  96 # X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.25, random_state = 42)
```

Logout

```
Python
 56
 57 # In[4]:
58
 60 le = LabelEncoder()
60 le = LabelEncoder()
df('salary_currency') = le.fit_transform(df['salary_currency'])
61 df('salary_currency') = le.fit_transform(df['employee_residence'])
62 df('company_location'] = le.fit_transform(df['company_location'])
63 df('company_size'] = le.fit_transform(df['company_size'])
64 df('company_size'] = le.fit_transform(df['experience_level'])
65 df('job_title'] = le.fit_transform(df['experience_level'])
 69 # In[5]:
 76 # In[6]:
 77
78
 79 # eda(df)
 80 # heat_map(df)
 81 # check_vif(df,['work_year','company_location','salary_currency'])
 84 # # Data splitting
 86 # In[7].
87 X = df. 1
      # Int():
X = df.loc[:, df.columns != 'salary_in_usd']
X = X.drop(['Unnamed: 0', 'work_year', 'salary', 'salary_currency', 'company_location'], axis = 1)
y = df['salary_in_usd']
 88
 91
92
91
2  # df = df.drop(['work_year', 'company_location', 'salary_currency', 'salary'], axis = 1)
93  # X = df.loc[', df.columns != 'salary_in_usd']
94  # X = X.drop(['Unnamed: 0', 'experience_level', 'employment_type', 'employee_residence', 'remote_ratio', 'company_size'], axis = 1)
95  # y = df['salary_in_usd']
 96 | # X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.25, random_state = 42)
 99 # In[8]:
 102 X. head()
105 # # Modeling
106
109
110 X_train, X_test, y_train, y_test = train_test_split(X,y,test_size = 0.25, random_state = 42)
111 lm = LinearRegression()
112 lm.fit(X_train, y_train)
113 y_pred = lm.predict(X_test)
114 mean_squared_error(y_test, y_pred)
116 # Im = LinearRegression()
110 # Im = Linearkegression()
117 # Im. fit(X_train, y_train)
118 # y_pred = Im. predict(X_test)
119 # mean_squared_error(y_test, y_pred)
120 # pickle.dump(lm, open('model.pkl', 'wb'))
123 # In[15]:
pickle.dump(lm, open('model.pkl','wb'))
flask_app = Flask(__name__)
model = pickle.load(open('model.pkl','rb'))
129 @flask_app.route('/')
130 def home():
                return render_template('flask_app.html')
int_features = [int(x) for x in request.form.values()]
final_features = [np.array(int_features)]
prediction = model.predict(final_features)
136
139
140
                 return render_template('flask_app.html', prediction_text = 'The salary will be $ {}'.format(prediction))
141
if __name__ == '__main__':
from waitress import serve
                 flask_app.run(port=5000, debug = True)
146
148
150
```

```
clDOCTYPE html>
chtml>
chtml
chtml>
chtml
chtml>
chtml
```