

DevInsight: Developer Salary Predictor & Explorer

Annexure – 1

Term Paper

Salary Prediction Using Machine Learning

A Term Paper Report

Submitted in partial fulfillment of the requirements for the award of degree of

Bachelor of Technology

(Computer Science and Engineering)

Submitted to



L LOVELY
P PROFESSIONAL
U UNIVERSITY

LOVELY PROFESSIONAL UNIVERSITY

PHAGWARA, PUNJAB

From 1st Feb 2024 to 25th April 2024

SUBMITTED BY

Name of the Student : Satwik Uppada

Registration No : 12111298

Faculty : Ajay Sharma Sir

Annexure – 11: Student Declaration

To whom so ever it may concern

I, Satwik Uppada, 12111298, hereby declare that the work done by me on “Salary Prediction based On Stack Overflow Developer Survey Data using Machine Learning” from Feb 2023 to April 2023, is a record of original work for the partial fulfilment of the requirements for the award of the degree, Bachelor of Technology.

Name of the Student : Satwik Uppada

Registration Number : 12111298

Dated : 13th April 2024

ACKNOWLEDGEMENT

Primarily I would like to thank god for being able to learn new technology. Then I would like to express my special thanks for gratitude to the teacher and instructor of the course Machine Learning who provided me the golden opportunity to learn a ne technology. I would like to also thank my own college Lovely Professional University for offering such a course Which not only improve my programming skill but also taught me other new technology. Then I would like to thank my parents and friends who have applied me their valuable suggesstions and guidance for choosing this course.

Finally, I would like to thank everyone who have helped me a lot.

Dated : 13th April 2024

Table of Contents

S. No.	Contents
1	Title
2	Student Declaration
3	Acknowledgement
4	Table of Contents
5	Abstract
6	Objective
7	Introduction
8	Theoretical Background
9	Hardware & Software
10	Methodology
11	Flowchart
12	Results
13	Summary
14	Conclusion

ABSTRACT

The "DevInsight" project dives into analyzing data from the 2023 Stack Overflow Developer Survey to understand how much developers earn. It looks at different factors like where they live, their education, and work experience. By cleaning and exploring the data, the project uncovers important trends. Then, using machine learning, it predicts salaries based on these factors. The project also creates a user-friendly website where people can interact with the data and get salary predictions. Overall, the goal is to help developers and companies make better decisions. The project uses standard computers and software like Python and Pandas to do this. It's all about making sense of data to help people in the tech industry. Through this project, users can gain valuable insights into the factors influencing developer salaries, empowering them to make informed decisions about their careers or hiring strategies. With its intuitive interface and accurate predictions, DevInsight simplifies the complexity of salary estimation in the tech industry, catering to both individual developers and businesses alike.

OBJECTIVE

The main objective of this project is to develop a web application, DevInsight, for predicting developer salaries based on various factors. Additionally, the project aims to explore the Stack Overflow Developer Survey data to gain insights into trends and patterns among developers worldwide. The application should provide accurate salary predictions using machine learning algorithms. Furthermore, it seeks to create an intuitive user interface for easy interaction and understanding of the prediction process. Lastly, the project aims to offer a seamless experience for users, from data exploration to salary prediction, enhancing their understanding of the developer job market. Moreover, the project aims to showcase proficiency in utilizing Python, Streamlit, and machine learning techniques for real-world data analysis tasks. It seeks to demonstrate effective data preprocessing, feature selection, model training, and evaluation processes. Ultimately, the objective is to provide a valuable tool for both developers seeking salary insights and data enthusiasts interested in exploring developer survey data.

INTRODUCTION

1. Background

The project comes from a simple idea: understanding how much developers earn and what factors influence their salaries can help both companies and developers make better decisions. This project is like a Swiss Army knife for data analysis, combining different tools and techniques to explore and predict developer salaries using real survey data.

It's a bit like looking at a treasure map – the survey data is like the map, and we're trying to find the hidden gems of information within it. By doing this, we're not just crunching numbers; we're unlocking valuable insights that can guide hiring strategies, career decisions, and industry trends.

In a nutshell, this project is about turning raw data into actionable insights. It's like solving a puzzle – each piece of data is like a puzzle piece, and by putting them together in the right way, we can see the bigger picture of what's happening in the developer world. And the best part? We're making all this complex stuff easy to understand and use for everyone.

2. What is Stack Overflow data

Stack Overflow data refers to the information collected from the annual Stack Overflow Developer Survey. Stack Overflow is a widely-used online platform where developers ask and answer questions related to programming and software development. The Stack Overflow Developer Survey collects comprehensive data on various aspects of developers' lives, including their demographics, education, job preferences, salary, coding habits, and more. This data provides valuable insights into the trends and preferences within the global developer community.

3. How they collected the data

The data for the Stack Overflow Developer Survey is collected through an annual online survey conducted by Stack Overflow. The survey is typically open to all developers worldwide and is promoted through various channels, including the Stack Overflow website, social media platforms, developer communities, and newsletters. Developers are invited to participate voluntarily, and the survey covers a wide range of topics related to their professional lives and experiences. The survey questions are designed to gather information about developers' demographics, employment status, programming languages, tools, technologies, and other relevant aspects.

4. Basic data information about the data

The basic data collected from the Stack Overflow Developer Survey includes various attributes related to developers' demographics, education, employment, and coding activities. Some of the key data points typically included in the survey dataset are:

- a. Developer demographics: Age, gender, country of residence.
- b. Education: Highest level of education completed, field of study.
- c. Employment: Employment status, company size, job role, years of professional experience.
- d. Coding activities: Programming languages used, coding frequency, preferred tools and technologies.
- e. Salary: Annual compensation, currency, benefits.
- f. In 2023 , The dataset comprises 84 columns, each representing a distinct aspect of the developer experience and industry trends.

THEORETICAL BACKGROUND

Machine Learning:

Machine learning is like teaching a computer to learn from data and make decisions or predictions without being explicitly programmed. It's like teaching a child how to ride a bicycle by showing them examples and letting them practice. Once the child learns, they can ride the bicycle on their own without constant instructions.

For example, imagine you want to build a model that predicts whether an email is spam or not. You would give the computer lots of examples of emails labeled as spam and not spam. The computer learns from these examples and develops a set of rules to distinguish between the two types of emails. Then, when you give it a new email, it can predict whether it's spam or not based on what it learned from the examples.

Exploratory Data Analysis (EDA):

Exploratory Data Analysis is like exploring a new city to understand its layout and features before deciding where to go. In data analysis, it's the process of examining and visualizing data to understand its characteristics, patterns, and relationships.

For example, let's say you have a dataset of housing prices. With EDA, you would look at things like the distribution of prices, the relationship between prices and features like the number of bedrooms or location, and any outliers or missing values. By visualizing the data with plots and charts, you can get a better sense of what the data looks like and what insights it might hold before diving into more detailed analysis or modeling.

Problem Statement

In today's tech-driven world, understanding the factors influencing developer salaries is crucial for both developers and employers. However, with a plethora of variables at play, predicting salaries accurately can be challenging. The problem statement revolves around developing a robust model that can accurately predict developer salaries based on various factors such as country, education level, and years of experience.

Algorithms Used

1. Linear Regression

Linear regression is a simple yet powerful algorithm used for predicting continuous values. It assumes a linear relationship between the independent variables and the target variable. Despite its simplicity, linear regression can provide valuable insights into how different features affect the target variable.

2. Decision Tree

Decision trees are versatile algorithms that can handle both regression and classification tasks. They partition the feature space into regions, making them intuitive to understand and interpret. Decision trees are particularly useful for capturing non-linear relationships and interactions between features.

3. Random Forest

Random Forest is an ensemble learning algorithm that constructs multiple decision trees and combines their predictions to improve accuracy and robustness. By averaging the predictions of multiple trees, Random Forest reduces overfitting and provides more reliable predictions.

4. GridSearchCV

GridSearchCV is a technique used for hyperparameter tuning, where it exhaustively searches through a specified parameter grid to find the optimal hyperparameters for a given model. By systematically exploring different parameter combinations, GridSearchCV helps optimize model performance.

Best Algorithm: Decision Tree

Among the algorithms tested, the Decision Tree model yielded the most accurate results for predicting developer salaries. Despite its simplicity, the Decision Tree algorithm was able to capture complex relationships between features and the target variable. Its ability to handle non-linear relationships and interactions made it well-suited for this task.

Benefits

- **Accuracy:** The Decision Tree model achieved high accuracy in predicting developer salaries, providing valuable insights into the factors influencing salary levels.
- **Interpretability:** Decision trees offer interpretability, allowing stakeholders to understand the decision-making process behind salary predictions.
- **Efficiency:** Decision trees are computationally efficient and can handle large datasets with ease, making them practical for real-world applications.
- **Flexibility:** Decision trees can handle both numerical and categorical data, making them versatile for a wide range of predictive modeling tasks.

Hardware & Software

The project requires standard hardware components such as a computer or laptop with sufficient processing power and memory. As for software, the following tools and libraries are utilized:

- **Python:** The primary programming language for data analysis, machine learning, and web development.
- **Pandas, Matplotlib, Seaborn:** Libraries for data manipulation and visualization.
- **Scikit-learn:** Library for implementing machine learning algorithms.
- **Streamlit:** Framework for building interactive web applications.
- **Pickle:** Library for serializing Python objects.

Methodology

1. Data Collection:

- **Source:** I obtained the data from the Stack Overflow Developer Survey, which provides valuable insights into the developer community. (<https://insights.stackoverflow.com/survey>)
- **Content:** The dataset contains information about developers' demographics, education, coding experience, employment status, and salary.

```
1 data = pd.read_csv("survey_results_public.csv")
2 data
```

2. Data Preprocessing:

- **Feature Selection:** I selected relevant features that could potentially influence developer salaries.

```
1 data = data[['Country', 'EdLevel', 'YearsCodePro', 'Employment', 'ConvertedCompYearly']]
2 data = data.rename({'ConvertedCompYearly': 'Salary'}, axis=1)
```

- **Cleaning:** I cleaned the dataset by removing irrelevant columns and handling missing values.

```
1 def shorten_categories(categories, cutoff):
2     categorical_map = {}
3     for i in range(len(categories)):
4         if categories.values[i] >= cutoff:
5             categorical_map[categories.index[i]] = categories.index[i]
6         else:
7             categorical_map[categories.index[i]] = 'Other'
8
9     return categorical_map
10
```

```

1 data = data[data["Salary"] <= 250000]
2 data = data[data["Salary"] >= 10000]
3 data = data[data['Country'] != 'Other']

```

```

1 country_map = shorten_categories(data.Country.value_counts(),400)
2 data['Country'] = data['Country'].map(country_map)
3 data.Country.value_counts()

```

```

1 def clean_experience(x):
2     if x == 'More than 50 years':
3         return 50
4     if x == 'Less than 1 year':
5         return 0.5
6     return float(x)
7
8 data['YearsCodePro'] = data['YearsCodePro'].apply(clean_experience)

```

- **Label Encoding:** Categorical variables like education level and country were encoded into numerical values for analysis.

```

1 from sklearn.preprocessing import LabelEncoder
2 le_education = LabelEncoder()
3 data['EdLevel'] = le_education.fit_transform(data['EdLevel'])
4 data["EdLevel"].unique()
5 #le.classes_

```

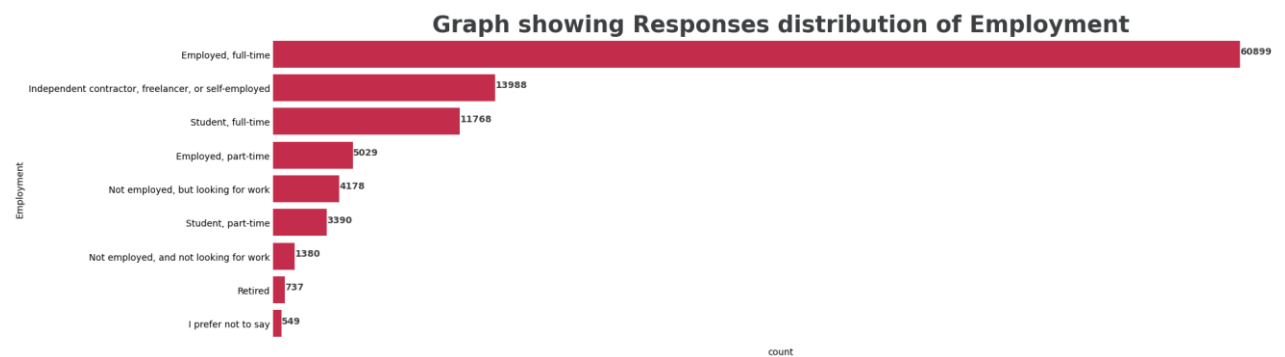
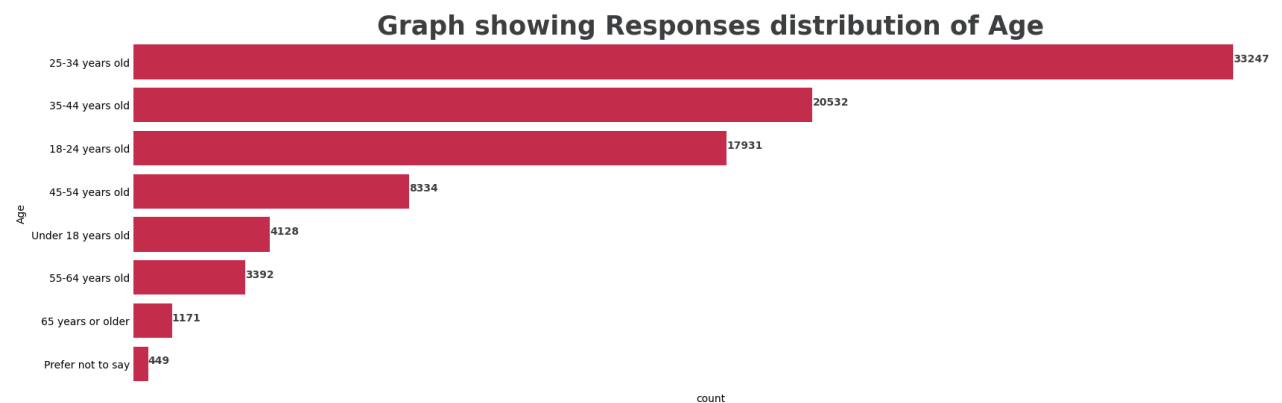
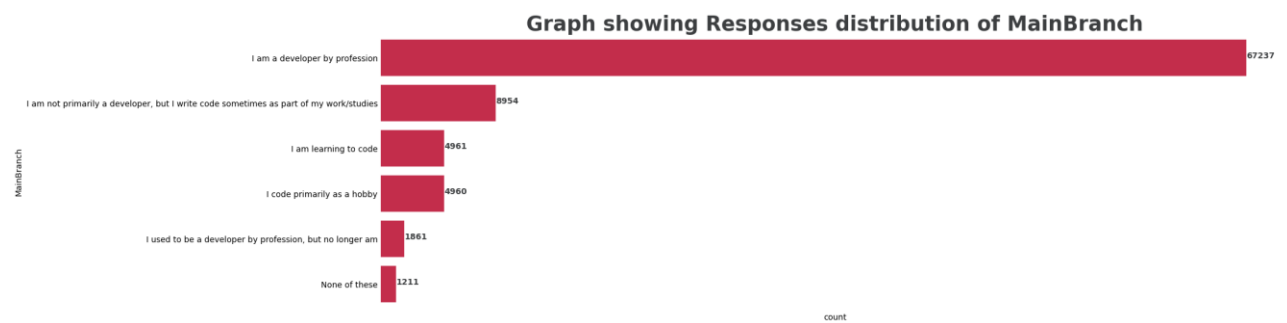
```

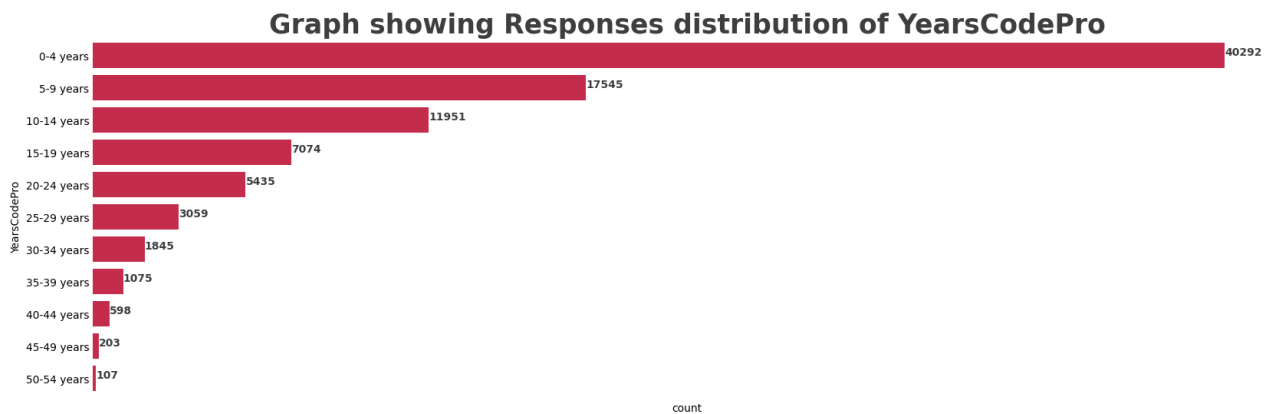
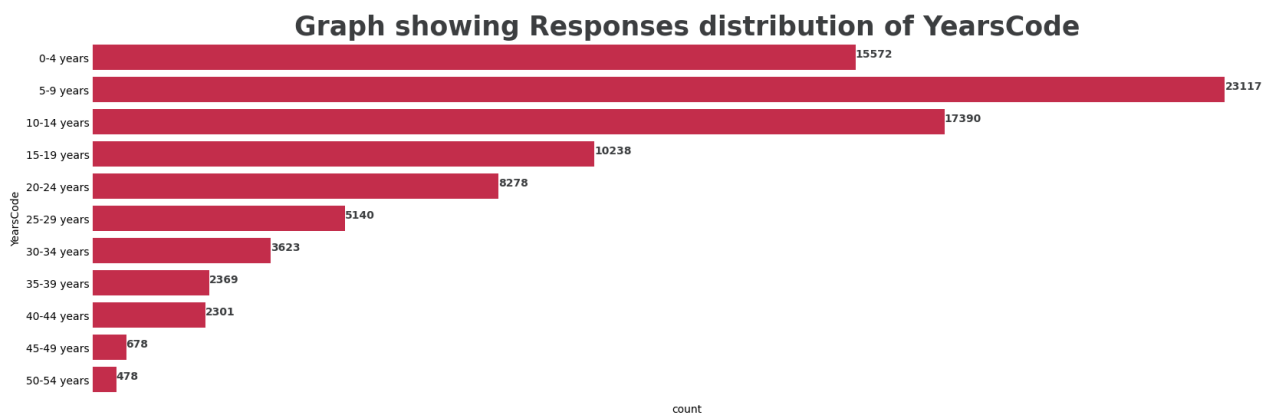
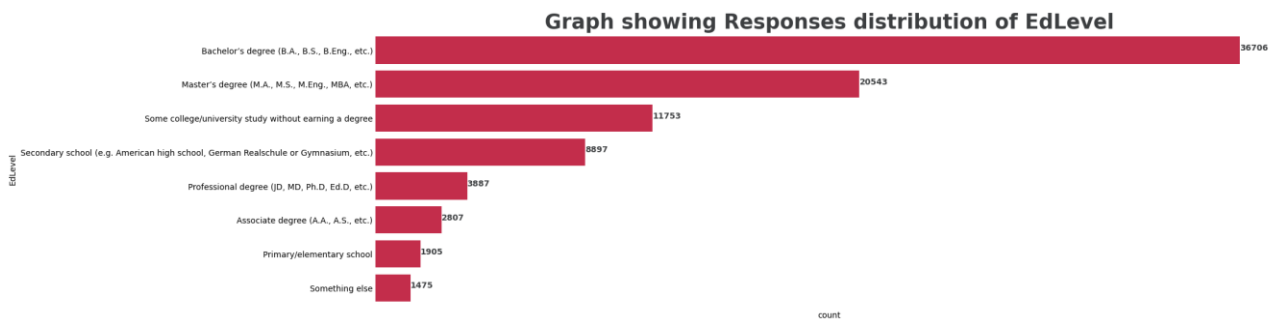
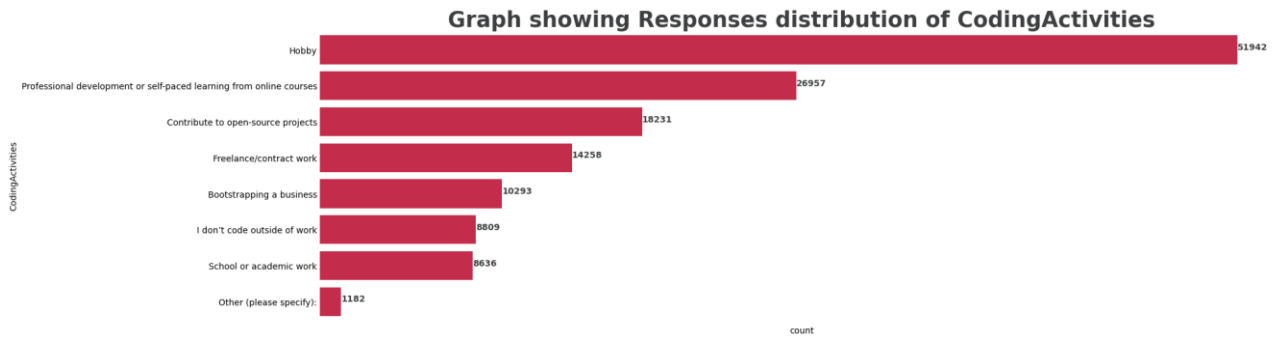
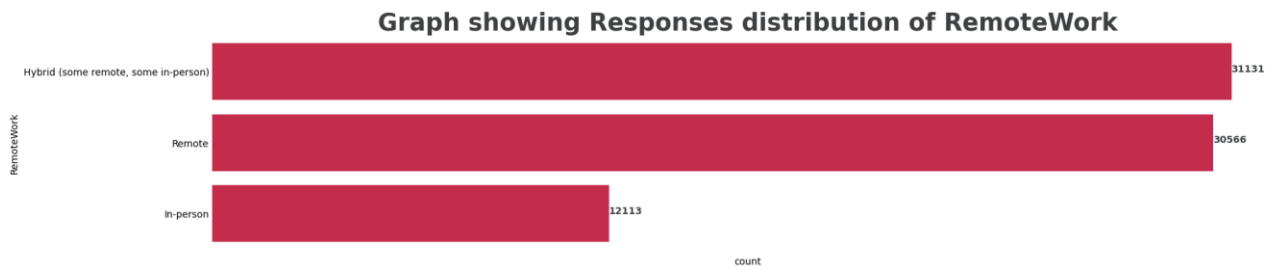
1 le_country = LabelEncoder()
2 data['Country'] = le_country.fit_transform(data['Country'])
3 data["Country"].unique()

```

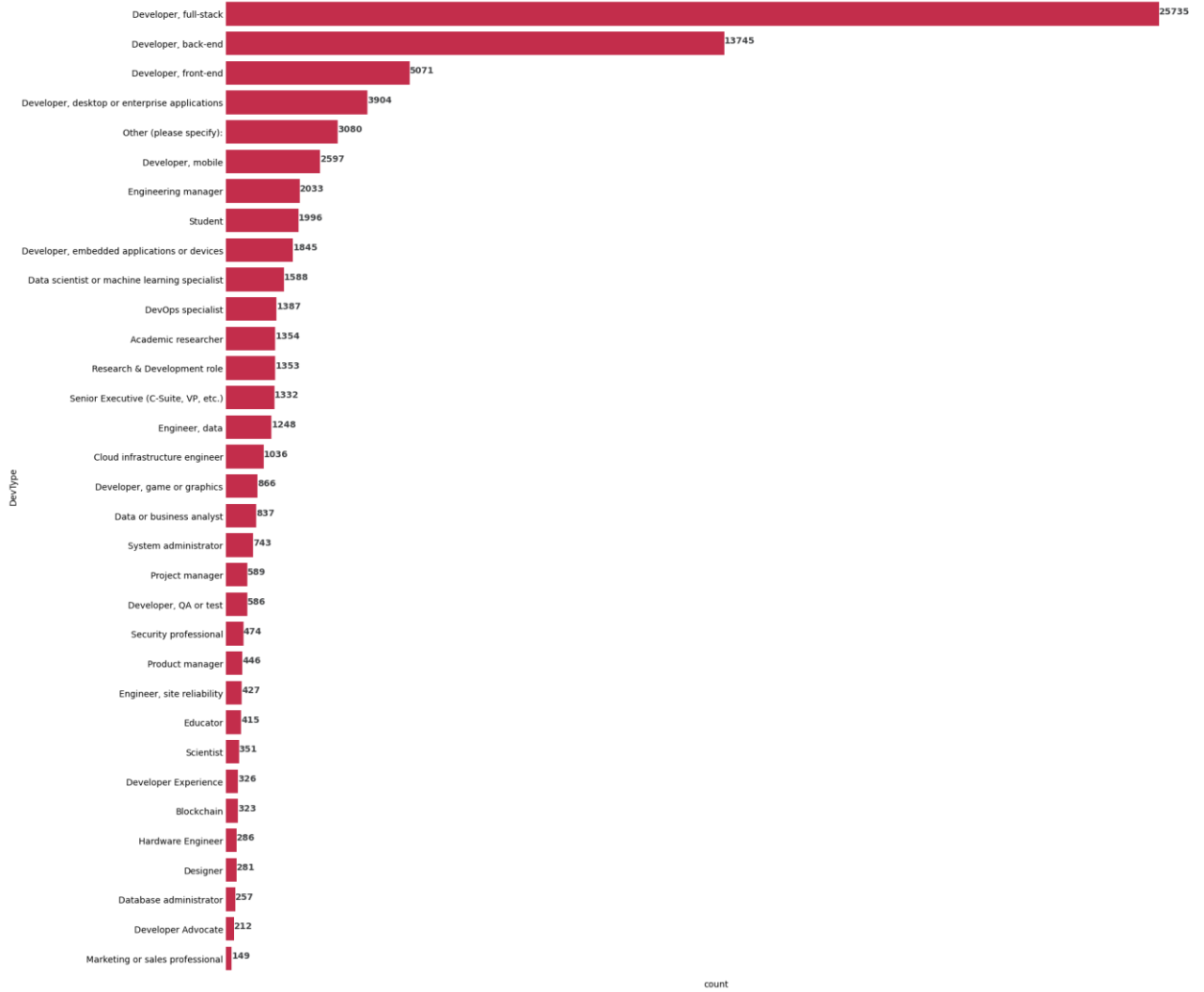
3. Exploratory Data Analysis (EDA):

- **Visualizations:** I explored the data through various visualizations such as bar charts and box plots to understand the distribution and relationships between different variables.
- **Insights:** EDA helped me identify patterns and trends in the data, such as the impact of education level and coding experience on salaries.

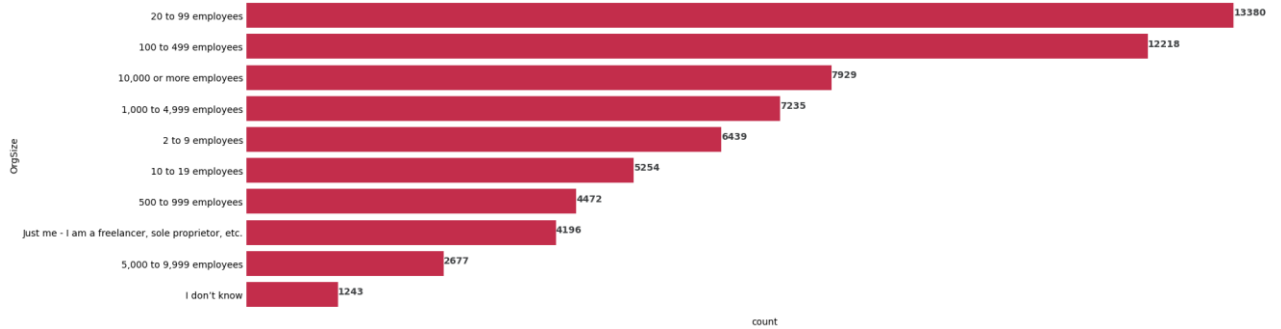




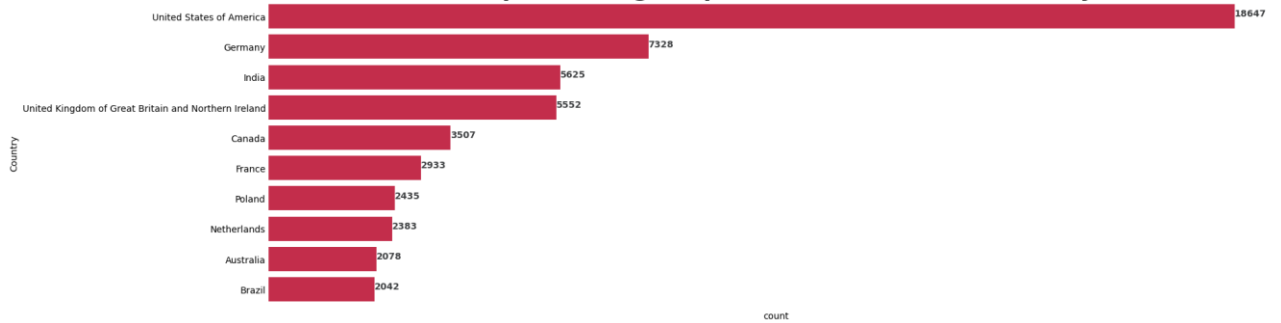
Graph showing Responses distribution of DevType



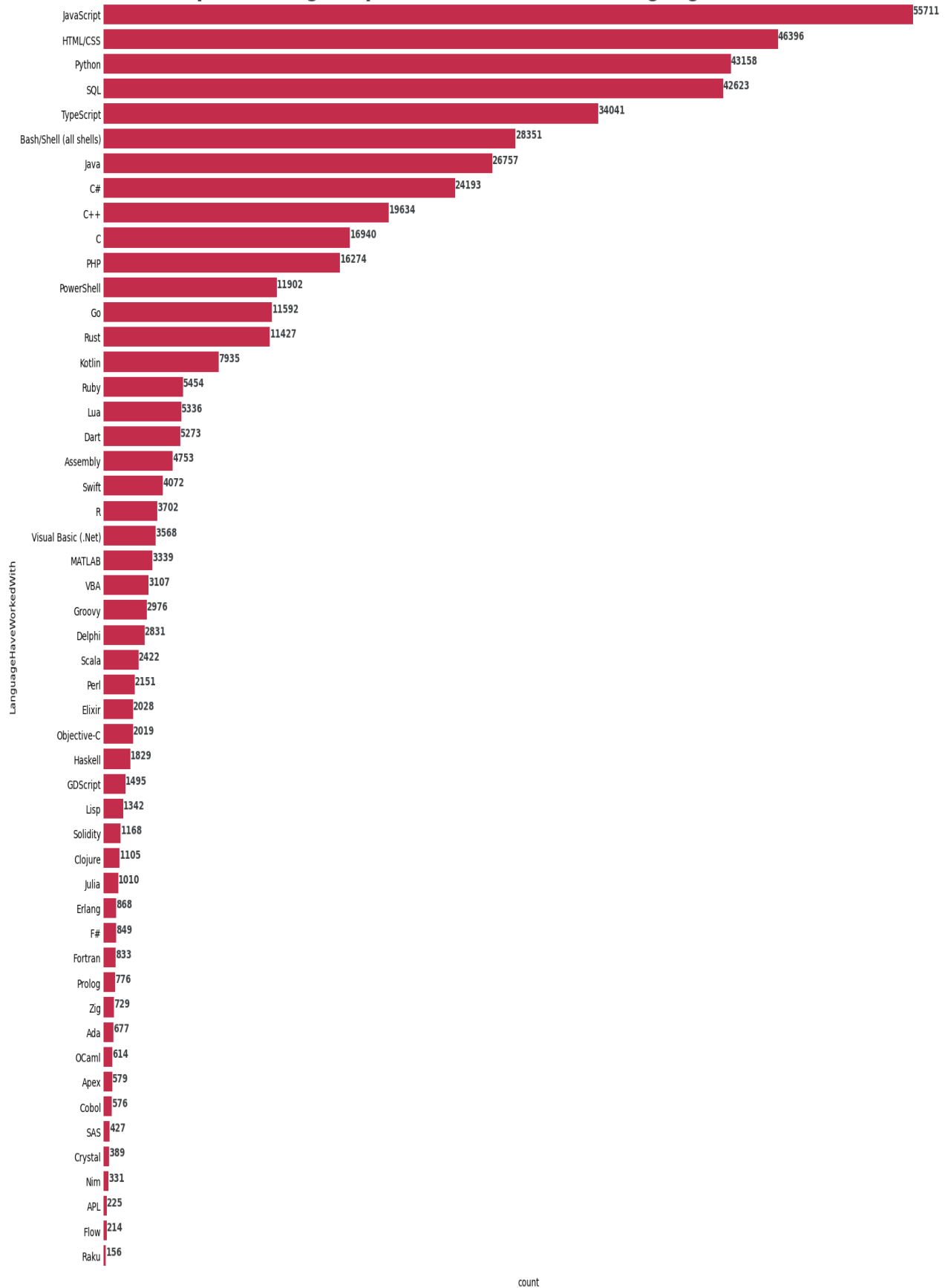
Graph showing Responses distribution of OrgSize



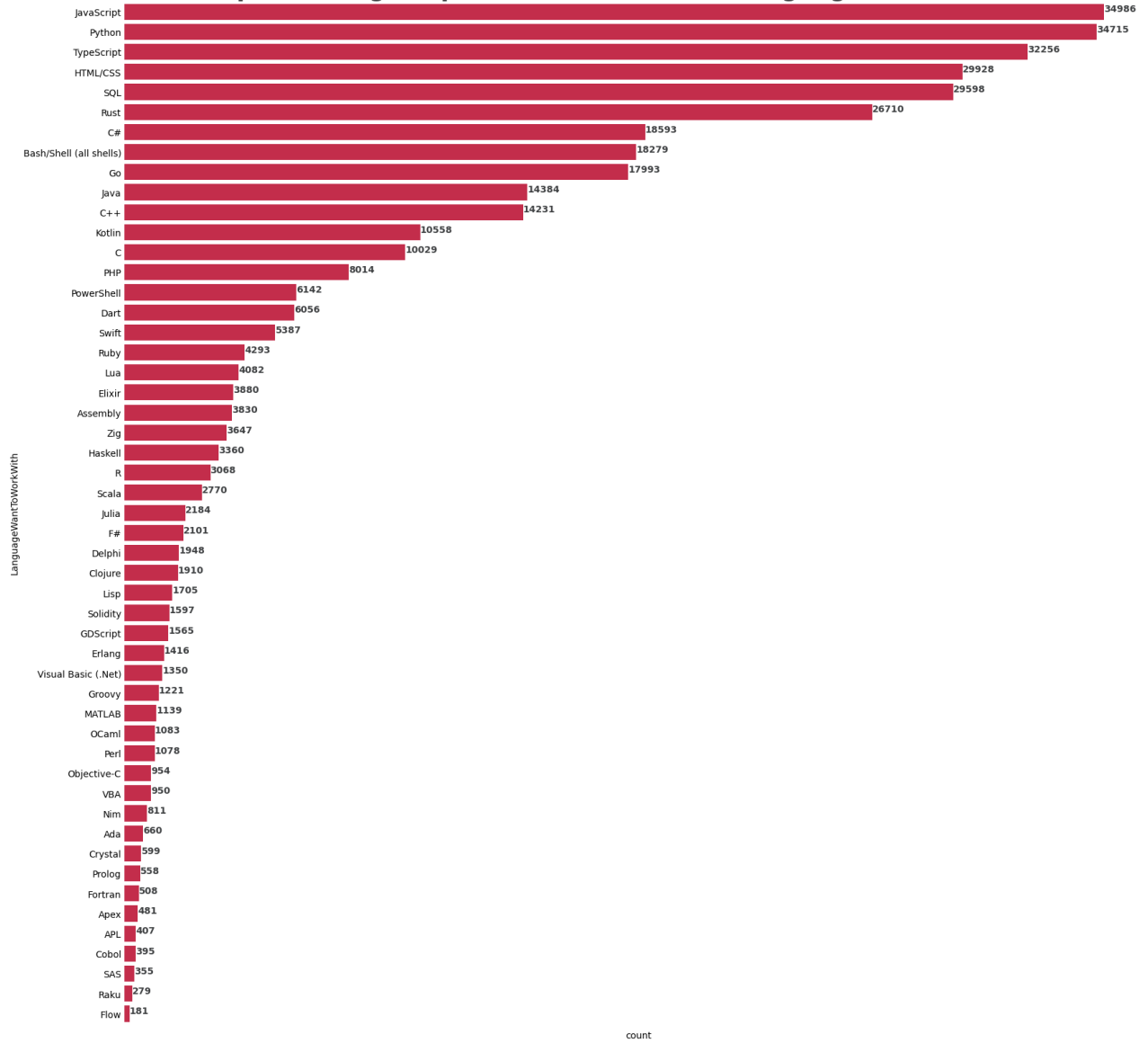
Graph showing Responses distribution of Country

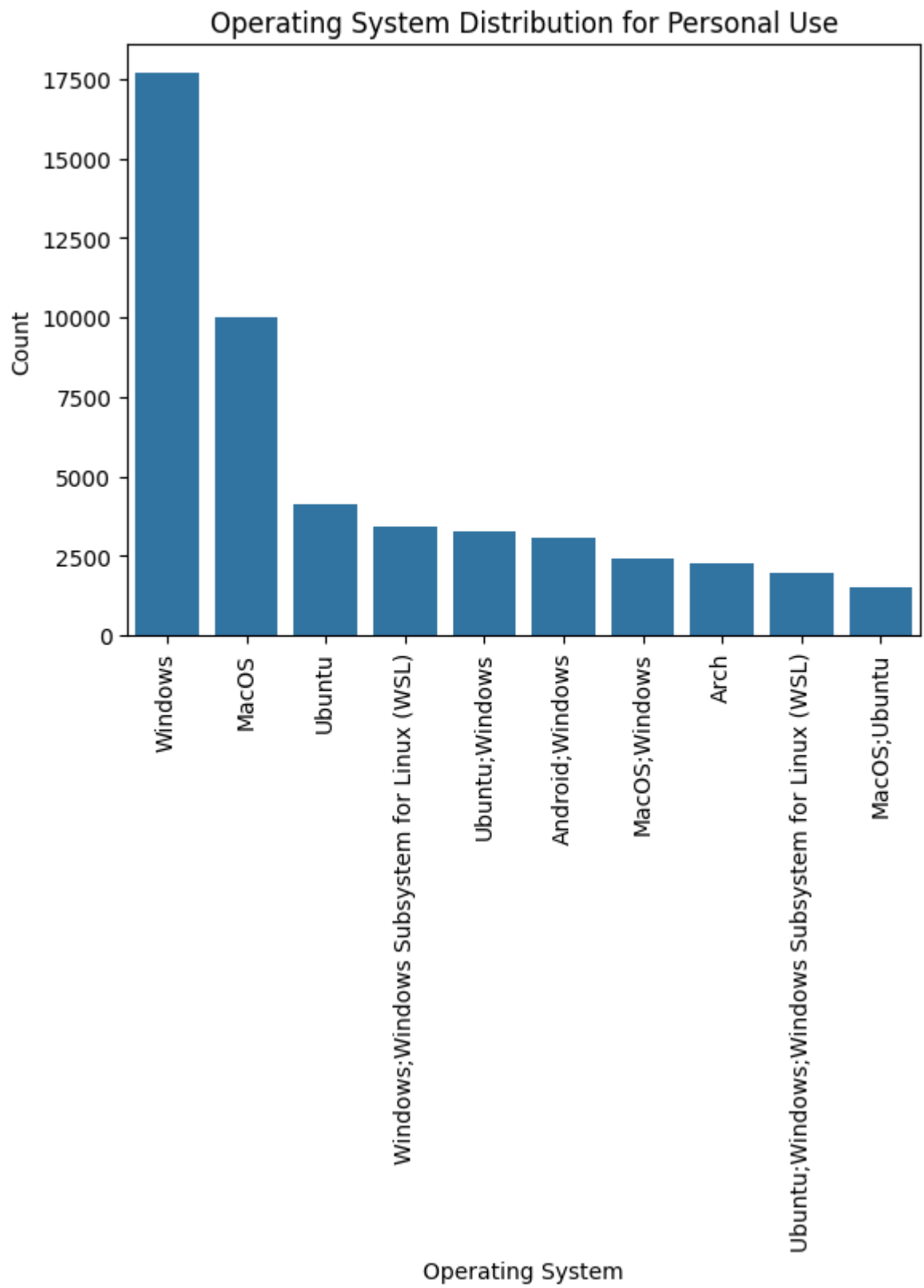


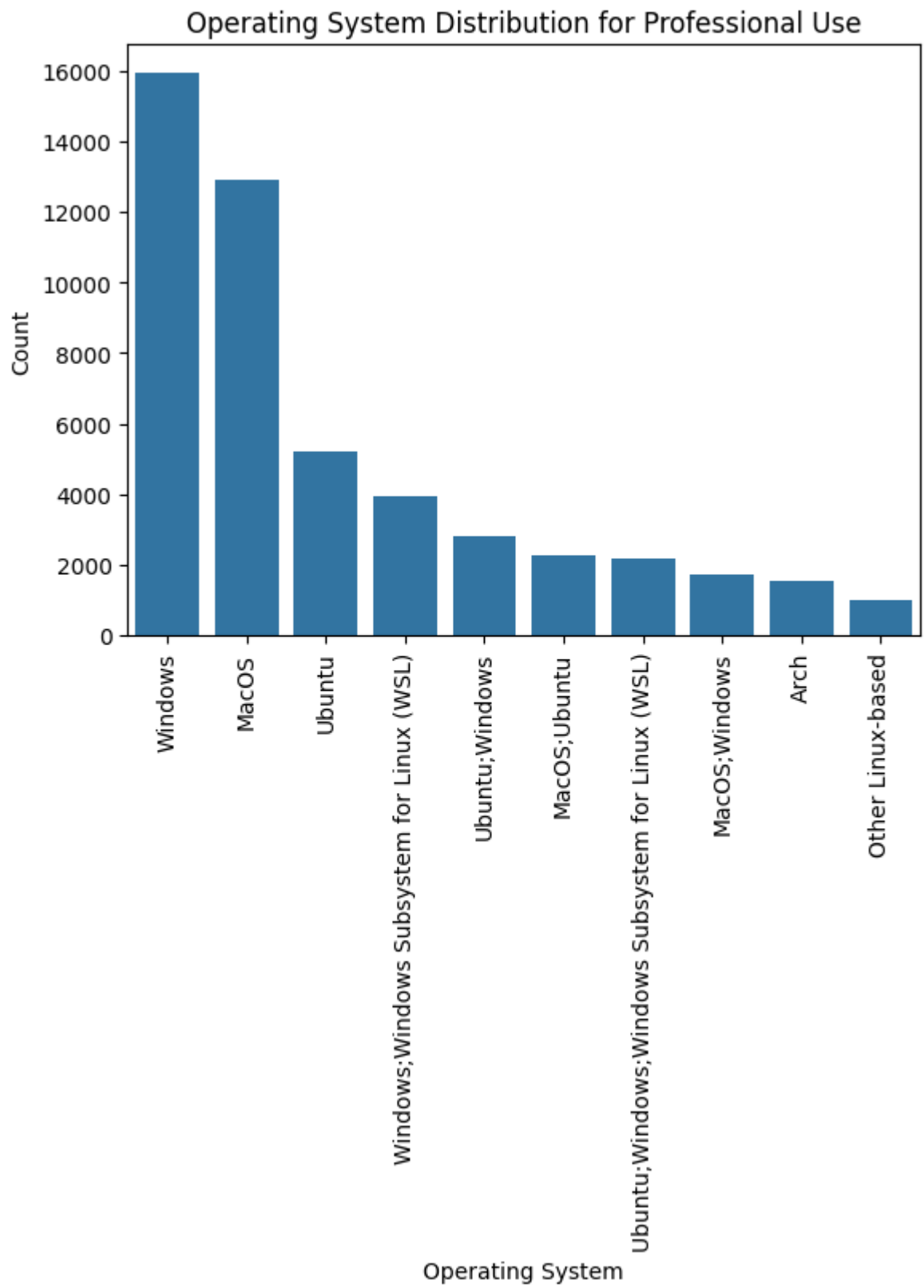
Graph showing Responses distribution of LanguageHaveWorkedWith



Graph showing Responses distribution of LanguageWantToWorkWith







4. Model Development:

- **Algorithms Used:** I employed machine learning algorithms including Linear Regression, Decision Tree, Random Forest, and GridSearchCV.

```
1 from sklearn.linear_model import LinearRegression
2 linear_reg = LinearRegression()
3 linear_reg.fit(X, y.values)
```

```
1 from sklearn.tree import DecisionTreeRegressor
2 dec_tree_reg = DecisionTreeRegressor(random_state=0)
3 dec_tree_reg.fit(X, y.values)
```

```
1 from sklearn.ensemble import RandomForestRegressor
2 random_forest_reg = RandomForestRegressor(random_state=0)
3 random_forest_reg.fit(X, y.values)
```

- **Training:** The models were trained on the cleaned and preprocessed data to learn the relationships between input features and salary.

```
1 y_pred = linear_reg.predict(X)
```

```
1 y_pred = dec_tree_reg.predict(X)
```

```
1 y_pred = random_forest_reg.predict(X)
```

- **Evaluation:** I evaluated the models' performance using metrics like Mean Squared Error (MSE) to measure their accuracy in predicting salaries.

```
1 from sklearn.metrics import  
    mean_squared_error, mean_absolute_error  
2 import numpy as np  
3 error = np.sqrt(mean_squared_error(y, y_pred))  
4 error
```

```
1 error = np.sqrt(mean_squared_error(y, y_pred))
2 print("${:,.02f}".format(error))
```

```
1 error = np.sqrt(mean_squared_error(y, y_pred))
2 print("${:,.02f}".format(error))
```

5. Model Selection:

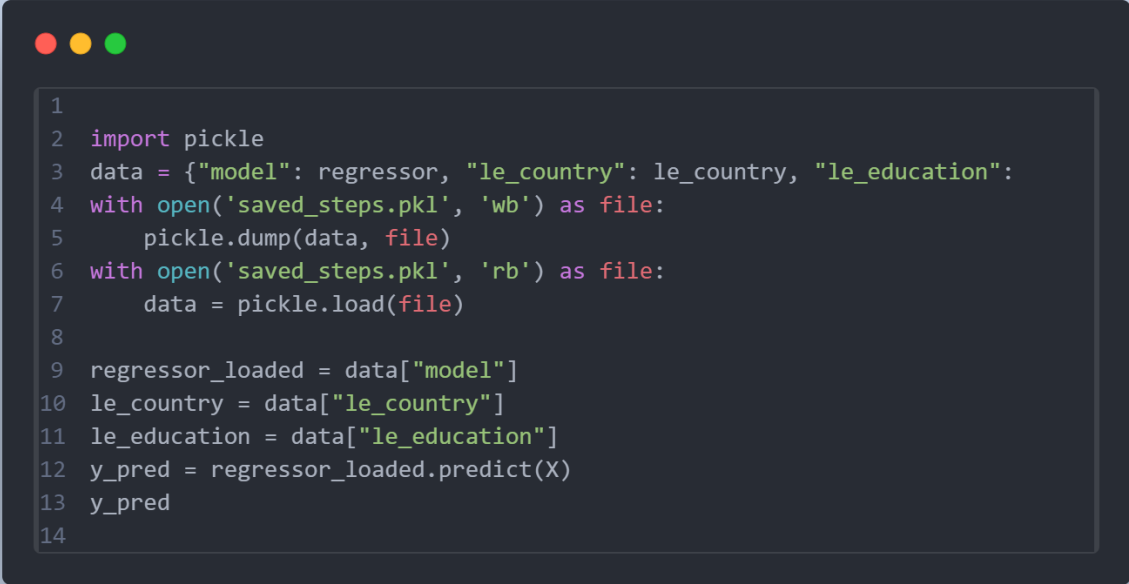
- **Comparison:** I compared the performance of different algorithms to identify the most accurate model for predicting developer salaries.

```
1 Model Performance Evaluation
2
3 1. Linear Regression:
4 Mean Squared Error (MSE): 46082.71
5 Description: Linear regression model's performance in predicting developer salaries.
6
7 2. Decision Tree:
8 Mean Squared Error (MSE): $33,748.85
9 Description: Evaluation of decision tree algorithm's accuracy in salary prediction.
10
11 3. Random Forest:
12 Mean Squared Error (MSE): $33,818.22
13 Description: Assessment of random forest model's performance in predicting developer salaries.
14
15 4. GridSearchCV:
16 Mean Squared Error (MSE): $34,647.64
17 Description: GridSearchCV algorithm's MSE score, indicating its predictive accuracy in salary estimation.
18
19 Best Performing Model:
20
21 Decision Tree:
22 Mean Squared Error (MSE): $33,748.85
23 Description: The decision tree algorithm exhibited the highest accuracy among all models,
24 making it the preferred choice for predicting developer salaries.
```

- **Selection:** Based on evaluation results, i chose the Decision Tree algorithm as it provided the most accurate predictions.

6. Model Deployment:

- **Saving:** I saved the trained Decision Tree model along with label encoders using pickle to deploy it for making predictions.



```
1
2 import pickle
3 data = {"model": regressor, "le_country": le_country, "le_education":
4 with open('saved_steps.pkl', 'wb') as file:
5     pickle.dump(data, file)
6 with open('saved_steps.pkl', 'rb') as file:
7     data = pickle.load(file)
8
9 regressor_loaded = data["model"]
10 le_country = data["le_country"]
11 le_education = data["le_education"]
12 y_pred = regressor_loaded.predict(X)
13 y_pred
14
```

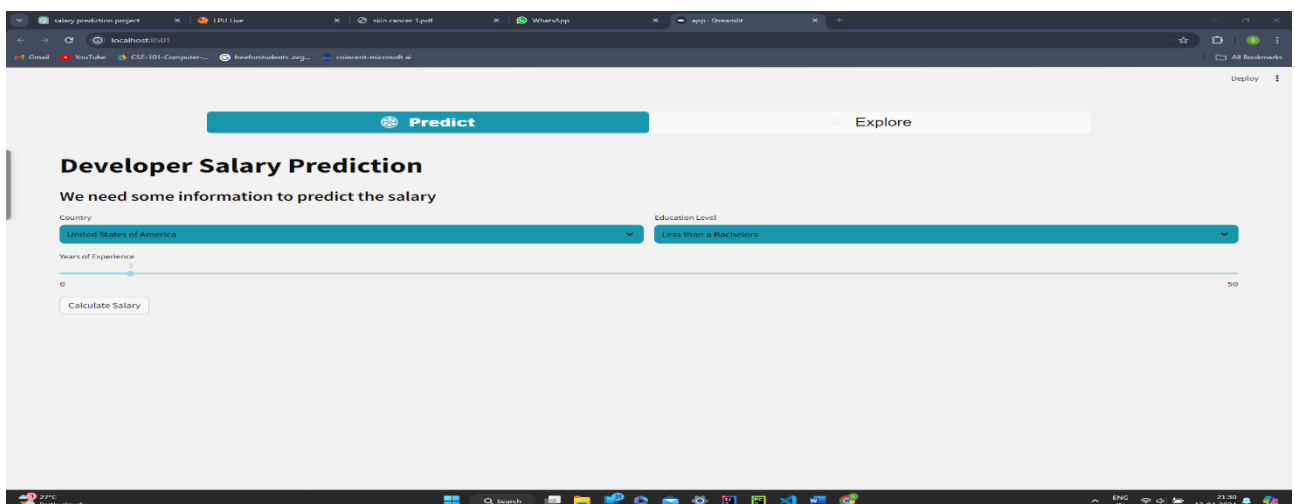
- **Web Application:** I developed a user-friendly web application using Streamlit framework to allow users to input their details and get salary predictions in real-time.

I created multipage application with streamlit. Here is the files.....

```

1 import streamlit as st
2 st.set_page_config(layout="wide")
3 from predict_page import show_predict_page
4 from explore_page import show_explore_page
5 from streamlit_option_menu import option_menu
6
7 # page = st.sidebar.selectbox("Explore Or Predict", ("Predict", "Explore"))
8
9 page = option_menu(
10     menu_title=None,
11     options=['Predict', 'Explore'],
12     icons=['fan'],
13     default_index=0,
14     orientation='horizontal',
15     styles={
16         "container": {"padding": "0!important",
17         "background-color": "#fafafa"},
18         "icon": {"color": "#f1f1f2", "font-size": "25px"},
19         "nav-link": {
20             "font-size": "25px",
21             "text-align": "center",
22             "margin": "0px",
23             "--hover-color": "#a1d6e2",
24         },
25         "nav-link-selected": {"background-color": "#1995ad"},
26     },
27 )
28
29 if page == "Predict":
30     show_predict_page()
31 else:
32     show_explore_page()

```



```

1 import streamlit as st
2 import pandas as pd
3 import matplotlib.pyplot as plt
4 import seaborn as sb
5
6
7 @st.cache_data
8 def load_data():
9     data = pd.read_csv("survey_results_public.csv")
10    data = data.drop(columns=['ResponseId', 'Q120', 'LearnCode', 'LearnCodeOnline', 'LearnCodeCoursesCert', 'PurchaseInfluence', 'TechList',
11    'Buy
12    'AIN
13    'AIBen'
14    'AINextVery different', 'AINextNeither different nor similar', 'AINextSomewhat similar', 'AINextVery similar',
15    'ICorPM', 'NEWSOSites', 'SOVisitFreq', 'SOAccount', 'SOPartFreq', 'SOComm', 'SOAI', 'AISelect', 'AISent', 'AIAcc',
16    'AIToolCurrently Using', 'AIToolNot interested in Using', 'Knowledge_1', 'Knowledge_2', 'Knowledge_3', 'Knowledge_4',
17    'Knowledge_8', 'Frequency_1', 'Frequency_2', 'Frequency_3', 'TimeSearching', 'TimeAnswering', 'SurveyLength',
18    'SurveyEa
19
20    return data
21
22 data = load_data()
23
24 def show_percentage(data, dataframe, column):
25     total_count = data[column].notnull().sum()
26     dataframe['percentage'] = round((dataframe['count'] / total_count) * 100, 2)
27     st.bar_chart(dataframe.set_index(column)['percentage'], use_container_width=True, color='#1995ad', height=500, width=70,)
28
29 def show_responses(dataframe, column):
30     total_count = data[column].notnull().sum()
31
32     st.bar_chart(dataframe.set_index(column), use_container_width=True, color='#1995ad', height=500, width=70)
33
34 def graph(data, dataframe, column, keyname):
35     col1, col2, col3, col4, col5, col6, col7, col8, col9, col10 = st.columns(10)
36
37     col1.success("{} responses".format(data[column].notnull().sum()))
38     if col10.button("Percentage", key="percentage_"+keyname, help='It will show percentage of people in each branch'):
39         show_percentage(data, dataframe, column)
40     if col9.button("Responses"):
41         show_responses(dataframe, column)
42     elif col9.button("Responses", key="response_"+keyname, help="It will show count of people in each branch"):
43         show_responses(dataframe, column)
44     else:
45         show_responses(dataframe, column)

```



```

1
2
3
4 def show_explore_page():
5     st.title("Look at the Survey Data")
6     pd.set_option("display.max_columns", None)
7     st.dataframe(data[1:], height=600)
8
9     st.write("<hr style='border: 2px dotted #1995ad'>", unsafe_allow_html=True)
10    st.title("Explore the 2023 StackOverFlow Survey Results")
11    st.write("<hr style='border: 2px solid #1995ad'>", unsafe_allow_html=True)
12    st.write(
13        """
14        ## Stack OverFlow Developer Survey 2023
15        """
16    )
17
18
19
20    #####
21    # Main Branches
22    st.write("### Main Branch")
23
24    MainBranch_Data = data['MainBranch'].value_counts(sort=True).reset_index()
25    graph(data, MainBranch_Data, 'MainBranch', 'mainbranch')
26
27    st.write("<hr>", unsafe_allow_html=True)
28    #####
29    st.write("### Age Groups")
30    Age_Data = data['Age'].value_counts().reset_index()
31    graph(data, Age_Data, 'Age', 'age')
32
33
34    st.write("<hr>", unsafe_allow_html=True)
35    #####
36    st.write("### Employment Type")
37    data['EmploymentType'] = data['Employment'].str.split(';').apply(lambda x: x if isinstance(x, list) else [])
38    df_exploded = data.explode('EmploymentType')
39    # Remove NaN values
40    df_exploded = df_exploded.dropna(subset=['EmploymentType'])
41
42    # Count the occurrences of each search engi
43    Employment_counts = df_exploded['EmploymentType'].value_counts().reset_index()
44    Employment_counts.columns = ['Employment', 'count']
45    graph(data, Employment_counts, 'Employment', 'Employee')
46
47    st.write("<hr>", unsafe_allow_html=True)
48    #####
49    st.write("### Work Environment")
50    Remote_work_data = data['RemoteWork'].value_counts().reset_index()
51    graph(data, Remote_work_data, 'RemoteWork', 'Remotework')
52
53    st.write("<hr>", unsafe_allow_html=True)
54    #####
55    st.write("### Coding Activities")
56    data['Coding'] = data['CodingActivities'].str.split(';').apply(lambda x: x if isinstance(x, list) else [])
57    df_exploded = data.explode('Coding')
58    # Remove NaN values
59    df_exploded = df_exploded.dropna(subset=['Coding'])
60
61    # Count the occurrences of each search engi
62    CodingActivity_counts = df_exploded['Coding'].value_counts().reset_index()
63    CodingActivity_counts.columns = ['CodingActivities', 'count']
64    graph(data, CodingActivity_counts, 'CodingActivities', 'codingact')
65
66    st.write("<hr>", unsafe_allow_html=True)
67    #####
68    st.write("### Education Level")
69    Edu_level_data = data['EdLevel'].value_counts().reset_index()
70    graph(data, Edu_level_data, 'EdLevel', 'edu')
71
72
73    st.write("<hr>", unsafe_allow_html=True)
74    #####
75    st.write("### Years of Coding Experience")
76    # Create a dictionary to map the text values to integers
77    text_to_int = {
78        "Less than 1 year": 1,
79        "More than 50 years": 50
80    }
81
82    # Convert the text values to integers using the dictionary
83    data.loc[data['YearsCode'].isin(text_to_int), "YearsCode"] = data.loc[data["YearsCode"].isin(text_to_int), "
84
85    # Convert the non-text values to integers
86    data["YearsCode"] = data["YearsCode"].fillna(0).astype(int)
87    ages = data['YearsCode']
88
89    # Define bin edges
90    bin_edges = list(range(0, 56, 5)) # Adjusted to include upper bounds
91    # Define bin Labels
92    bin_labels = [f"{i}-{i+4} years" for i in range(0, 55, 5)] # Adjusted to have one Less Label
93    # Cut the ages into bins
94    age_bins = pd.cut(ages, bins=bin_edges, labels=bin_labels, right=False)
95    # Count occurrences within each bin
96    age_counts = age_bins.value_counts().sort_index().reset_index()
97    graph(data, age_counts, 'YearsCode', 'expcode')
98
99    st.write("<hr>", unsafe_allow_html=True)
100    #####
101    st.write("### Years of Experience as a Professional")
102    # Create a dictionary to map the text values to integers
103    text_to_int = {
104        "Less than 1 year": 1,
105        "More than 50 years": 50
106    }
107
108    # Convert the text values to integers using the dictionary
109    data.loc[data['YearsCodePro'].isin(text_to_int), "YearsCodePro"] = data.loc[data["YearsCodePro"].isin(text_t
110
111    # Convert the non-text values to integers
112    data["YearsCodePro"] = data["YearsCodePro"].fillna(0).astype(int)
113    ages = data['YearsCodePro']
114
115    # Define bin edges
116    bin_edges = list(range(0, 56, 5)) # Adjusted to include upper bounds
117    # Define bin Labels
118    bin_labels = [f"{i}-{i+4} years" for i in range(0, 55, 5)] # Adjusted to have one Less Label
119    # Cut the ages into bins
120    age_bins = pd.cut(ages, bins=bin_edges, labels=bin_labels, right=False)
121    # Count occurrences within each bin
122    age_counts = age_bins.value_counts().sort_index().reset_index()
123    graph(data, age_counts, 'YearsCodePro', 'expcodepro')
124
125
126    st.write("<hr>", unsafe_allow_html=True)
127    #####
128
129
130
131

```


salary prediction project

LPU Live

skin cancer 1.pdf

WhatsApp

app - Streamlit

localhost:8501

Gmail

YouTube

CSE-101-Computer...

freefontstudents.org...

coincert-microsoft ai

Deploy

Predict

Explore

Look at the Survey Data

	MainBranch	Age	Employment	RemoteWork	CodingActivities	EdLevel	YearCode	
1	I am a developer by profession	25-34 years old	Employed, full-time	Remote	Hobby;Contribute to open-source projects;Bootstrapping a business;Professional de	Bachelor's degree (B.A., B.S., B.Eng., etc.)	18	9
2	I am a developer by profession	45-54 years old	Employed, full-time	Hybrid (some remote, some in-person)	Hobby;Professional development or self-paced learning from online courses	Bachelor's degree (B.A., B.S., B.Eng., etc.)	27	23
3	I am a developer by profession	25-34 years old	Employed, full-time	Hybrid (some remote, some in-person)	Hobby	Bachelor's degree (B.A., B.S., B.Eng., etc.)	12	7
4	I am a developer by profession	25-34 years old	Employed, full-time;independent con	Remote	Hobby;Contribute to open-source projects;Professional development or self-paced le	Bachelor's degree (B.A., B.S., B.Eng., etc.)	6	4
5	I am a developer by profession	35-44 years old	Employed, full-time	Remote	Hobby;Professional development or self-paced learning from online courses	Some college/university study without earning a degree	21	21
6	I am a developer by profession	35-44 years old	Employed, full-time	Remote	Hobby;Contribute to open-source projects;Professional development or self-paced le	Some college/university study without earning a degree	4	3
7	I am a developer by profession	25-34 years old	Employed, full-time	Remote	Hobby	Bachelor's degree (B.A., B.S., B.Eng., etc.)	5	3
8	I am not primarily a developer, t	45-54 years old	Employed, full-time	Hybrid (some remote, some in-person)	Hobby;Contribute to open-source projects	Master's degree (M.A., M.S., M.Eng., MBA, etc.)	20	15
9	I am a developer by profession	25-34 years old	Not employed, but looking for work	None	None	Bachelor's degree (B.A., B.S., B.Eng., etc.)	6	Hi
10	I am a developer by profession	25-34 years old	Employed, full-time	Remote	Hobby	Bachelor's degree (B.A., B.S., B.Eng., etc.)	14	3
11	I am a developer by profession	25-34 years old	Employed, full-time	Remote	Hobby	Bachelor's degree (B.A., B.S., B.Eng., etc.)	10	9
12	I am a developer by profession	25-34 years old	Employed, full-time	Remote	Hobby;Professional development or self-paced learning from online courses	Bachelor's degree (B.A., B.S., B.Eng., etc.)	10	9
13	I am not primarily a developer, t	35-44 years old	Employed, full-time	Hybrid (some remote, some in-person)	Hobby;Professional development or self-paced learning from online courses	Master's degree (M.A., M.S., M.Eng., MBA, etc.)	5	Le
14	I am a developer by profession	25-34 years old	Employed, full-time	Hybrid (some remote, some in-person)	Hobby	Master's degree (M.A., M.S., M.Eng., MBA, etc.)	15	7
15	I am a developer by profession	35-44 years old	Employed, full-time	Hybrid (some remote, some in-person)	Hobby;Contribute to open-source projects;Professional development or self-paced le	Master's degree (M.A., M.S., M.Eng., MBA, etc.)	15	10
16	I am a developer by profession	35-44 years old	Employed, full-time	Hybrid (some remote, some in-person)	Hobby	Bachelor's degree (B.A., B.S., B.Eng., etc.)	4	2

27°C
Partly cloudy

Search

ENG
IN

21:27
13-04-2024

salary prediction project

LPU Live

skin cancer 1.pdf

WhatsApp

app - Streamlit

localhost:8501

Gmail

YouTube

CSE-101-Computer...

freefontstudents.org...

coincert-microsoft ai

Deploy

Explore the 2023 StackOverFlow Survey Results

Stack Overflow Developer Survey 2023

Main Branch

89184 responses

Responses

Percentage

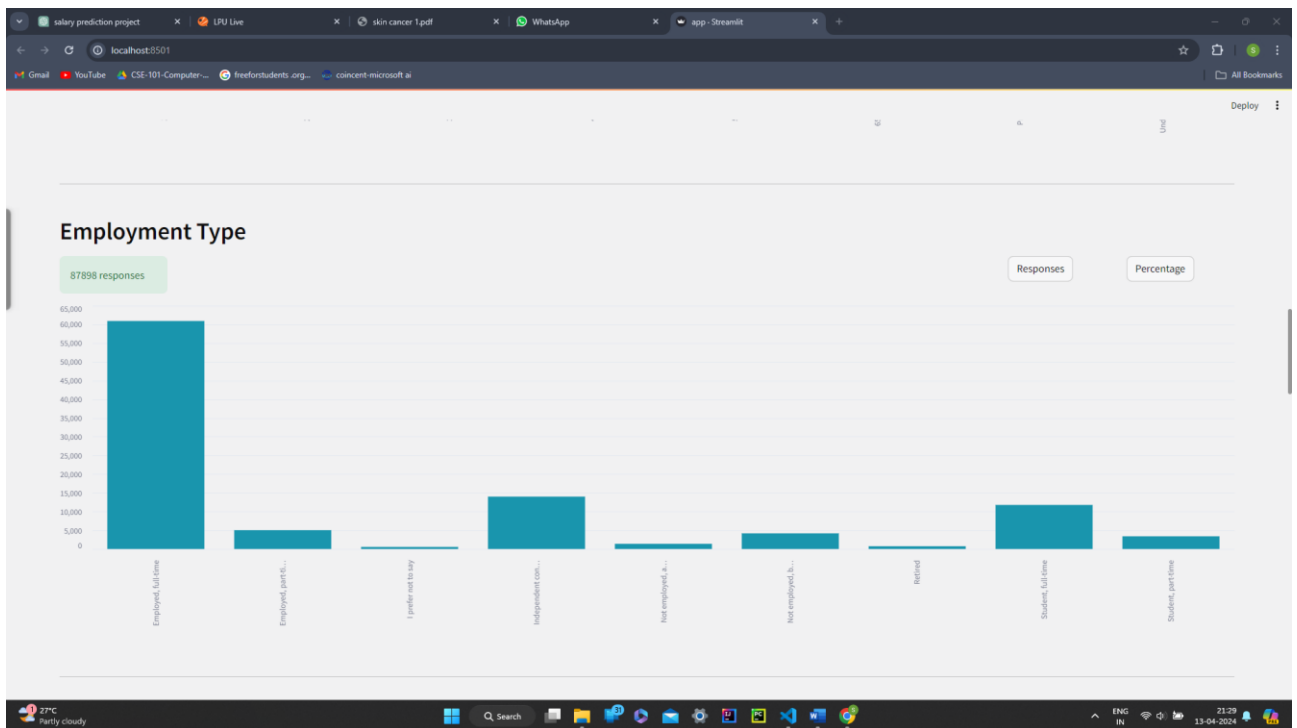
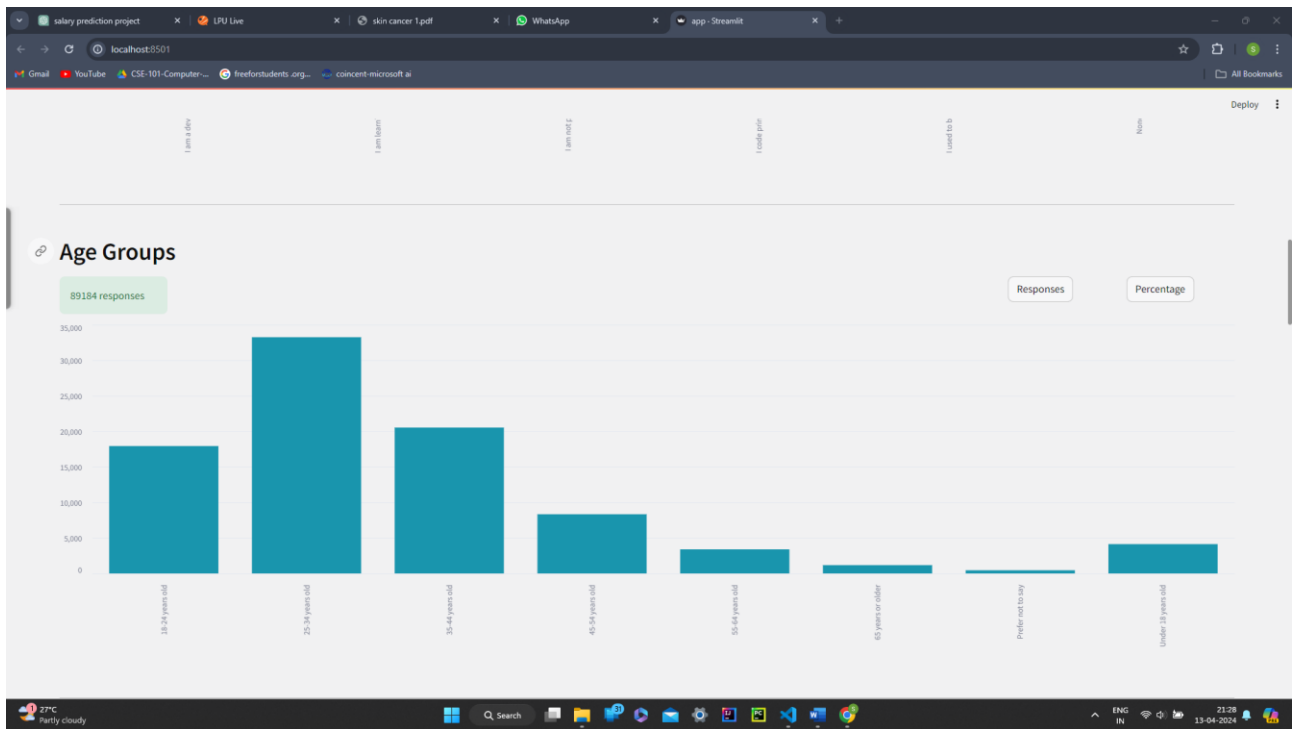
Main Branch	Responses
I am a developer by profession	89184
I am learning to become a developer	~5,000
I am not primarily a developer	~10,000
I consider myself a developer	~10,000
I intend to become a developer	~5,000
None of these	~5,000

27°C
Partly cloudy

Search

ENG
IN

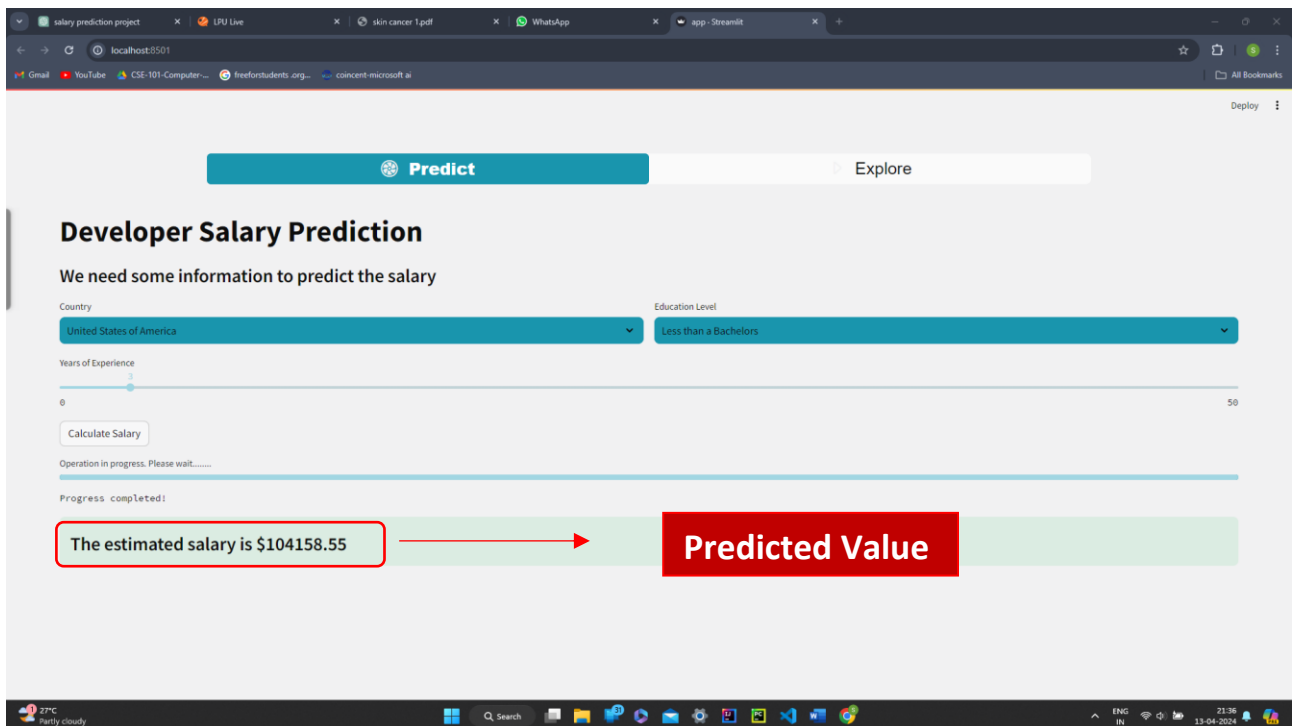
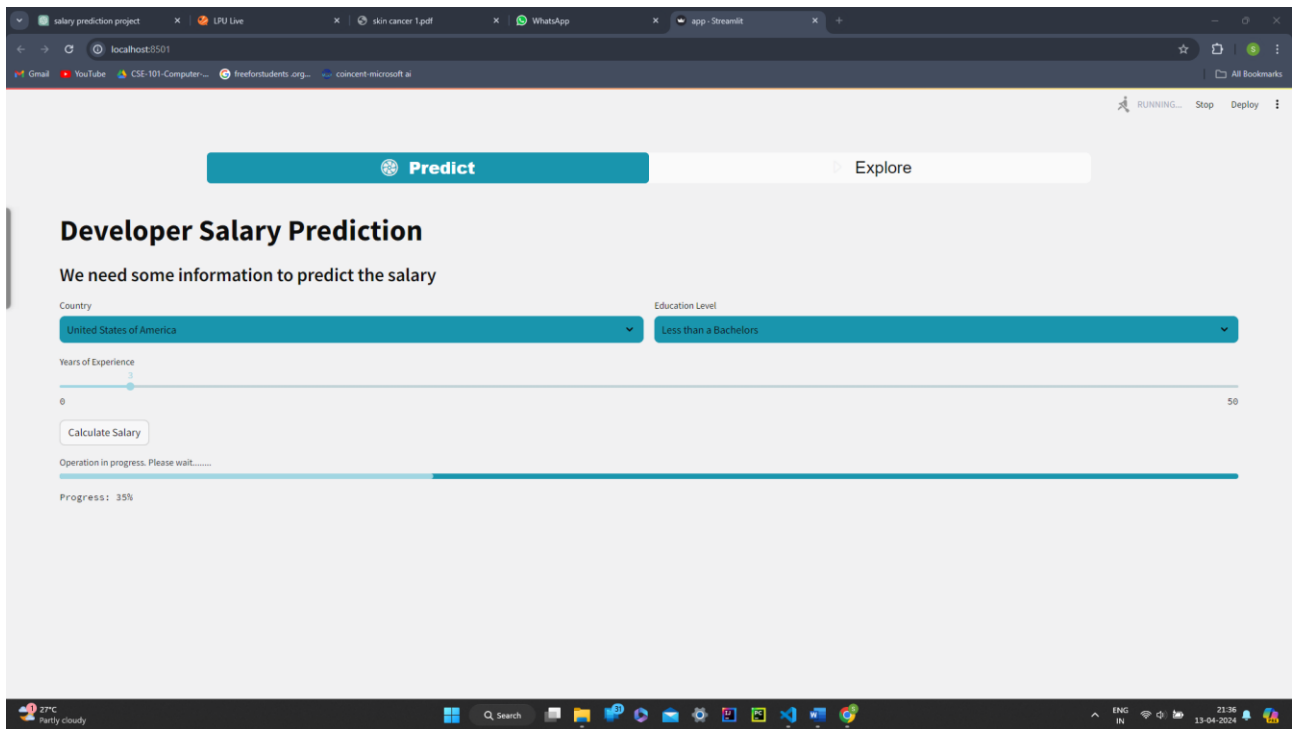
21:27
13-04-2024



```

1 import streamlit as st
2 import pickle
3 import numpy as np
4 import time
5
6
7 def load_model():
8     with open('saved_steps.pkl', 'rb') as file:
9         data = pickle.load(file)
10    return data
11
12 data = load_model()
13
14 regressor = data["model"]
15 le_country = data["le_country"]
16 le_education = data["le_education"]
17
18 def show_predict_page():
19     st.title("Developer Salary Prediction")
20
21     st.write("""### We need some information to predict the salary""")
22
23     countries = (
24         'United States of America'
25         ,
26         'Germany',
27         'United Kingdom of Great Britain and Northern Ireland',
28         'Canada',
29         'India',
30         'France',
31         'Netherlands',
32         'Australia',
33         'Brazil',
34         'Spain',
35         'Sweden',
36         'Italy',
37         'Poland',
38         'Switzerland',
39         'Denmark',
40         'Norway',
41         'Israel',
42         'Other'
43     )
44
45     education = (
46         "Less than a Bachelors",
47         "Bachelor's degree",
48         "Master's degree",
49         "Post grad",
50     )
51
52     col1,col2 = st.columns(2)
53     country = col1.selectbox("Country", countries)
54     education = col2.selectbox("Education Level", education)
55
56     expericence = st.slider("Years of Experience", 0, 50, 3)
57
58     ok = st.button("Calculate Salary",help="Hit me for prediction")
59     if ok:
60         progress_text = "Operation in progress. Please wait....."
61         progress_bar = st.progress(0)
62         status_text = st.empty()
63
64         for percent_complete in range(101):
65             # Update progress bar
66             progress_bar.progress(percent_complete,text=progress_text)
67
68             # Update status text
69             status_text.text(f'Progress: {percent_complete}%)
70
71             # Add a delay to simulate work being done
72             time.sleep(0.1)
73
74             # After completion, replace progress bar with a message
75             status_text.text('Progress completed!')
76
77             # spinner
78             with st.spinner('Wait for it...'):
79                 time.sleep(5)
80
81             # st.success('Done!')
82
83             X = np.array([[country, education, expericence ]])
84             X[:, 0] = le_country.transform(X[:,0])
85             X[:, 1] = le_education.transform(X[:,1])
86             X = X.astype(float)
87
88             salary = regressor.predict(X)
89             st.success(f"### The estimated salary is ${salary[0]:.2f}")

```



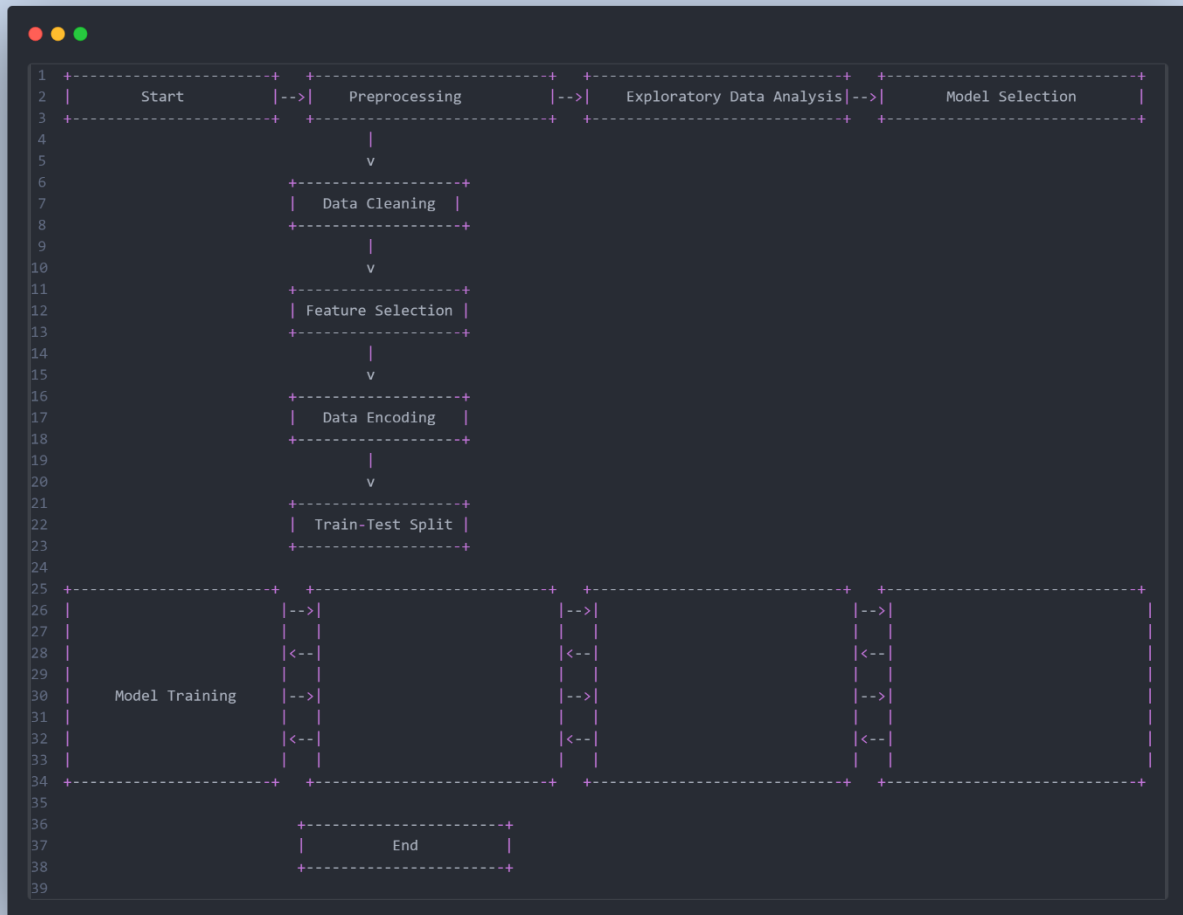
7. Testing and Validation:

- **Testing:** I tested the deployed model with sample inputs to ensure its functionality and accuracy.
- **Validation:** The model's predictions were validated against known salary data to verify its reliability and effectiveness.

8. Optimization and Refinement:

- **Fine-tuning:** I fine-tuned the model parameters using GridSearchCV to optimize its performance further.
- **Feedback Loop:** I incorporated feedback from testing and validation to refine the model and improve its accuracy over time.

Flowchart



Results

- **Linear Regression:**
 - Mean Squared Error (MSE): \$46,082.71
- **Decision Tree:**
 - MSE: \$33,748.85
- **Random Forest:**
 - MSE: \$33,818.22
- **GridSearchCV:**
 - MSE: \$34,647.64

The Decision Tree and Random Forest models demonstrated lower MSE values compared to Linear Regression and GridSearchCV, indicating better predictive performance.

Summary

- Explored and preprocessed the Stack Overflow developer survey data.
- Utilized machine learning algorithms like Linear Regression, Decision Tree, Random Forest, and GridSearchCV for salary prediction.
- Developed a Streamlit web application for both data exploration and salary prediction.
- Achieved accurate results comparable to Stack Overflow's data.

You can check same results in Stack Overflow results [here](#).

- Model.sav file created using the best-performing algorithm for deployment.

Conclusion

The project successfully addressed the challenge of predicting developer salaries using Stack Overflow survey data. By employing various machine learning algorithms and thorough preprocessing, we were able to develop accurate prediction models. The Streamlit web application provides an intuitive interface for users to explore survey data and obtain salary predictions based on their input. This project enhances our understanding of developer demographics and salary trends, offering valuable insights for stakeholders in the tech industry.