Group members:
2461: Bhaktee Ugale
2469: Vidhi Vaswani

Batch: B4

Problem Statement

Designing the Machine Learning Model for predicting salary based on parameters such as country, years of experience, educational Qualification etc using decision tree regressor.

Introduction
Tittle: Salary Prediction Model
We Have Used Linear Regressor and Decision Tree Regressor To Predict The Salary

Linear Regression:Linear Regression is a machine learning algorithm based on supervised learning. It performs a regression task. Regression models a target prediction value based on independent variables

Decision Tree Regressor: It's used to solve regression problems.Decision tree builds regression models in the form of a tree structure

Dataset mentioned is used for predicting salary. First we created a regressor model and saved it in a pickle file. We used that pickle file to predict salary. We also visualized the data in the form of piechart, line graph and bar graph. For frontend we used streamlit which is used for machine learning.

Data Set Information:

1. Data set used: StackOverflow survey 2021
2. Parameters considered while predicting salary: country, educational qualification, years of experience.
3. Few data samples:

	Country	EdLevel	YearsCodePro	Salary
9	Sweden	Master's degree	4.0	51552.0
11	Spain	Bachelor's degree	5.0	46482.0
12	Germany	Master's degree	6.0	77290.0
16	Turkey	Bachelor's degree	2.0	17748.0
17	Canada	Bachelor's degree	6.0	46135.0

Code and Output:

Regressor model code:

```
#importing necessarary libraries and reading csv file
import pandas as pd
import matplotlib.pyplot as plt

df=pd.read_csv("survey_results_public.csv")

#selecting attributes
df=df[["Country", "EdLevel", "YearsCodePro", "Employment",
"ConvertedCompYearly"]]
df=df.rename({"ConvertedCompYearly":"Salary"}, axis=1)
# data cleaning - removing data with null values
df=df[df["Salary"].notnull()]
df=df[df["YearsCodePro"].notnull()]
df=df[df["Country"].notnull()]
df=df[df["EdLevel"].notnull()]
df=df[df["Employment"].notnull()]
#data cleaning
df=df.dropna()
df=df[df["Employment"]=="Employed full-time"]
df=df.drop("Employment", axis=1)
def shorten_categories(categories, cutoff):
    categorical_map={}
    for i in range(len(categories)):
        if categories.values[i]>=cutoff:
```

```
            categorical_map[categories.index[i]] = categories.index[i]
        else:
            categorical_map[categories.index[i]] = 'Other'
    return categorical_map
# changing the country name to other whose developer/SD count is less than
400
country_map = shorten_categories(df.Country.value_counts(), 400)
df["Country"] = df["Country"].map(country_map)
df=df[df["Salary"]>=10000]
df=df[df["Salary"]<=250000]
df=df[df["Country"]!='Other']
def clean_experience(x):
    if x=="More than 50 years":
        return 50
    if x=='Less than 1 year':
        return 0.5
    return float(x)

df['YearsCodePro']=df['YearsCodePro'].apply(clean_experience)
def clean_education(x):
    if x=="Bachelor's degree (B.A., B.S., B.Eng., etc.)":
        return 'Bachelor’s degree'
    if x=="Master's degree (M.A., M.S., M.Eng., MBA, etc":
        return 'Master’s degree'
    if "Professional degree" in x or "Other doctoral" in x:
        return 'Post grad'
    return 'Less Than a Bachelor’s'

df['EdLevel']=df['EdLevel'].apply(clean_education)
#Transforming Data
# as edlevel , country and devtype are strings we convert them to numbers
from sklearn.preprocessing import LabelEncoder
le_education=LabelEncoder()
df['EdLevel']=le_education.fit_transform(df['EdLevel'])
df['EdLevel'].unique()

#LabelEncoder for country
le_country=LabelEncoder()
df['Country']=le_country.fit_transform(df['Country'])
```

```
#Splitting The Data
X=df.drop("Salary",axis=1)
y=df["Salary"]

#creating decision tree regressor object
from sklearn.tree import DecisionTreeRegressor
dec_tree_reg=DecisionTreeRegressor(random_state=0)
dec_tree_reg.fit(X,y.values)
y_pred=dec_tree_reg.predict(X)
error=np.sqrt(mean_squared_error(y,y_pred))
print("${:,.02f}".format(error))
from sklearn.model_selection import GridSearchCV
max_depth=[None,2,4,6,8,10,12]
parameters={"max_depth":max_depth}

#applying decision tree regressor to predict salary
regressor=DecisionTreeRegressor(random_state=0)
gs=GridSearchCV(regressor,parameters,scoring="neg_mean_squared_error")
gs.fit(X,y.values)
regressor=gs.best_estimator_
regressor.fit(X,y.values)
y_pred=regressor.predict(X)
error=np.sqrt(mean_squared_error(y,y_pred))
print("${:,.02f}".format(error))

#creating pickle file
import pickle
data={"model":regressor,"le_country":le_country,"le_education":le_educatio
n}
with open('saved_steps.pkl','wb') as file:
  pickle.dump(data,file)
```

app.py

```
import streamlit as st
from predict_page import show_predict_page
from explore_page import show_explore_page

page = st.sidebar.selectbox("Explore Or Predict", ("Predict", "Explore"))
```

```
if page=="Predict":
    show_predict_page()
else:
    show_explore_page()
```

predict_page.py

```
import streamlit as st
import pickle
import numpy as np


def load_model():
    with open('saved_steps.pkl','rb') as file:
        data=pickle.load(file)
    return data

data=load_model()

regressor=data["model"]
le_country=data["le_country"]
le_education=data["le_education"]


def show_predict_page():
    st.title("Software Developer Salary Prediction")
    st.header('''Enter Information to determine Salary''')

    country={
        "United States of America",
        "India",
        "Germany",
        "United Kingdom of Great Britain and Northern Ireland",
        "Canada",
        "France",
        "Brazil",
        "Spain",
        "Netherlands",
        "Australia",
        "Poland",
```

```
        "Italy",
        "Russian Federation ",
        "Sweden",
        "Turkey",
        "Switzerland",
        "Israel",
        "Norway"
    }

    education={
        "Master's degree",
        "Bachelor's degree",
        "Post grad",
       "Less Than a Bachelor's"
    }

    countries = st.selectbox("Country", country)
    edulevel = st.selectbox("Country", education)
    experience = st.slider("Years of Experience", 0, 50, 3)

    ok = st.button("Calculate Salary")
    if ok:
        X=np.array([[countries, edulevel, experience]])
        X[:,0]=le_country.transform(X[:,0])
        X[:,1]=le_education.transform(X[:,1])
        X=X.astype(float)

        salary = regressor.predict(X)

        st.subheader(f"The estimated salary is ${salary[0]:.2f}")
```

explore_page.py

```
import streamlit as st
import pandas as pd
import matplotlib.pyplot as plt

def shorten_categories(categories, cutoff):
    categorical_map={}
```

```
    for i in range(len(categories)):
        if categories.values[i]>=cutoff:
            categorical_map[categories.index[i]] = categories.index[i]
        else:
            categorical_map[categories.index[i]] = 'Other'
    return categorical_map


def clean_experience(x):
    if x=="More than 50 years":
        return 50
    if x=='Less than 1 year':
        return 0.5
    return float(x)

def clean_education(x):
    if x=="Bachelor's degree (B.A., B.S., B.Eng., etc.)":
        return 'Bachelor’s degree'
    if x=="Master's degree (M.A., M.S., M.Eng., MBA, etc":
        return 'Master’s degree'
    if "Professional degree" in x or "Other doctoral" in x:
        return 'Post grad'
    return 'Less Than a Bachelor’s'

@st.cache
def load_data():
    df=pd.read_csv("survey_results_public.csv")
    df=df[["Country", "EdLevel", "YearsCodePro", "Employment",
"ConvertedCompYearly"]]
    df=df.rename({"ConvertedCompYearly":"Salary"}, axis=1)
    df=df[df["Salary"].notnull()]
    df=df[df["YearsCodePro"].notnull()]
    df=df[df["Country"].notnull()]
    df=df[df["EdLevel"].notnull()]
    df=df[df["Employment"].notnull()]
    df=df.dropna()
    df=df[df["Employment"]=="Employed full-time"]
    df=df.drop("Employment", axis=1)
    country_map = shorten_categories(df.Country.value_counts(), 400)
    df["Country"] = df["Country"].map(country_map)
```

```
    df=df[df["Salary"]>=10000]
    df=df[df["Salary"]<=250000]
    df=df[df["Country"]!='Other']
    df['YearsCodePro']=df['YearsCodePro'].apply(clean_experience)
    df['EdLevel']=df['EdLevel'].apply(clean_education)
    return df

df = load_data()

def show_explore_page():
    st.title("Explore Software Engineer Salaries")

    st.header("Stack Overflow Developer Survey 2021")

    data=df["Country"].value_counts()

    fig1, ax1 = plt.subplots()
    ax1.pie(data, labels=data.index, autopct="%1.1f%%", startangle=90)
    ax1.axis("equal")

    st.subheader("Number of Data from different Countries")

    st.pyplot(fig1)

    st.subheader("Mean Salary Based On Country")

    data =
df.groupby(["Country"])["Salary"].mean().sort_values(ascending=True)
    st.bar_chart(data)

    st.subheader("Mean Salary Based On Experience")

    data =
df.groupby(["YearsCodePro"])["Salary"].mean().sort_values(ascending=True)
    st.line_chart(data)
```

Input:

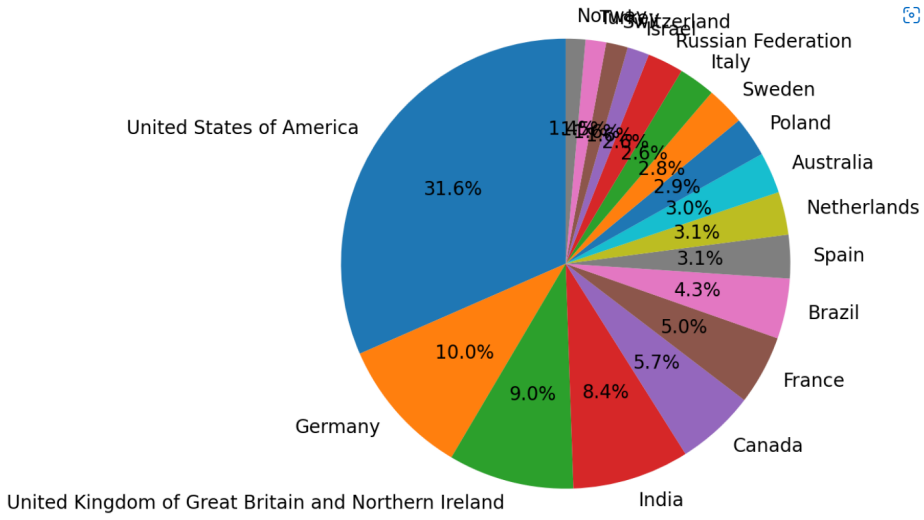
Country: Turkey

Educational Level: Bachelor’s Degree
Years of experience: 5

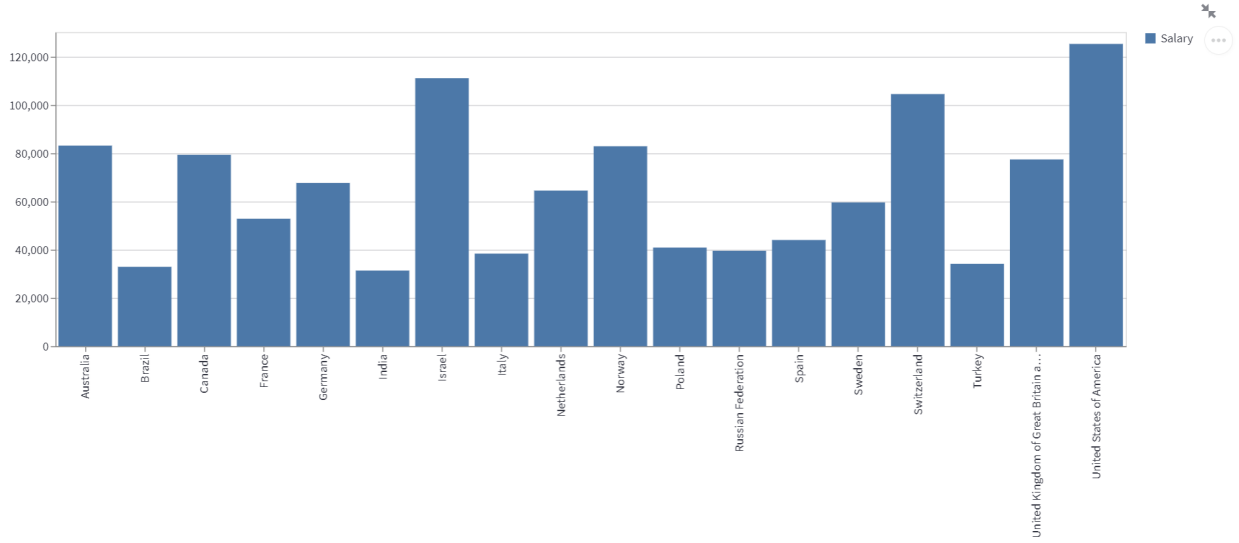
Output:

Data visualization:

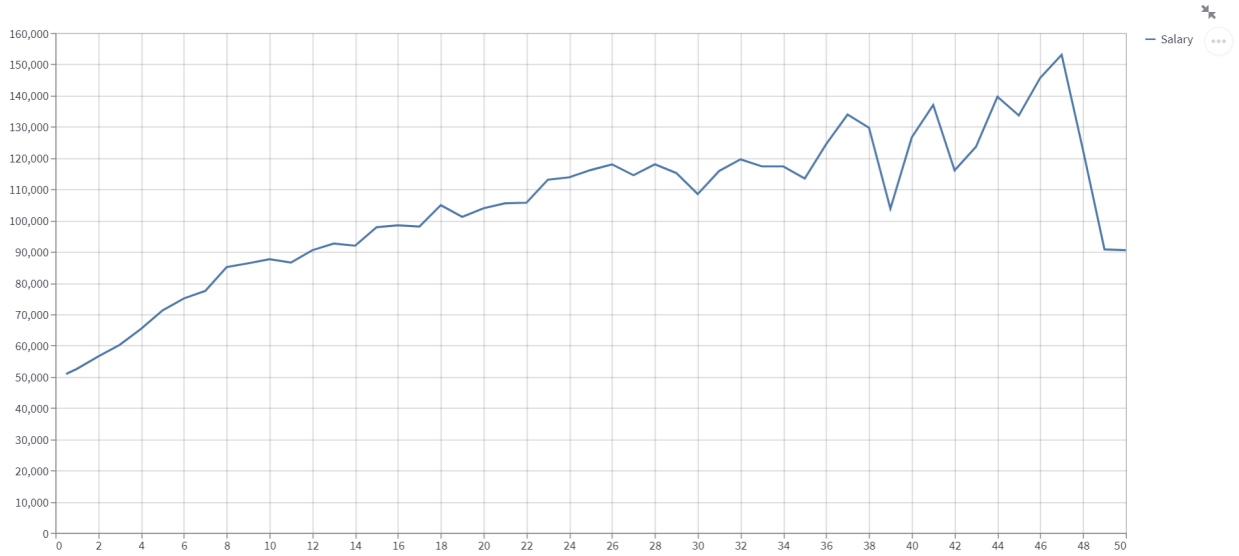
Pie chart of Distribution of Data from different Countries



Mean Salary Based On Country



Mean Salary Based On Experience



Conclusion:

1. Two types of regression models used for predicting salary out of which decision tree regressor gave minimum error so decision tree regressor is used.

2. Decision tree regressor implemented in machine learning project to predict salary using python.

References:

1) https://docs.streamlit.io/
2) https://docs.python.org/3/library/pickle.html
3)https://www.geeksforgeeks.org/python-decision-tree-regression-using-sklearn/
4)https://www.mygreatlearning.com/blog/gridsearchcv/