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

import pandas as pd
import seaborn as sns
import numpy as np
import matplotlib.pyplot as plt
from sklearn import linear_model
regr = linear_model.LinearRegression()
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
from sklearn.model_selection import cross_val_score
from sklearn.metrics import r2_score
#we have to upload ds_salaries file in sample_data
file=pd.read_csv("/content/sample_data/ds_salaries.csv")
data=file

data.head(10)
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call d
```

	work_year	experience_level	employment_type	job_title	salary	salary_curre
0	2023	SE	FT	Principal Data Scientist	80000	I
1	2023	MI	CT	ML Engineer	30000	U
2	2023	MI	CT	ML Engineer	25500	U
3	2023	SE	FT	Data Scientist	175000	U
4	2023	SE	FT	Data Scientist	120000	U
5	2023	SE	FT	Applied Scientist	222200	U
6	2023	SE	FT	Applied Scientist	136000	U
7	2023	SE	FT	Data Scientist	219000	U
8	2023	SE	FT	Data Scientist	141000	U
9	2023	SE	FT	Data Scientist	147100	U

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3755 entries, 0 to 3754
Data columns (total 11 columns):
 #   Column              Non-Null Count  Dtype 
---  ------              --------------  ----- 
 0   work_year           3755 non-null   int64 
 1   experience_level    3755 non-null   object
 2   employment_type     3755 non-null   object
 3   job_title           3755 non-null   object
 4   salary              3755 non-null   int64 
 5   salary_currency     3755 non-null   object
 6   salary_in_usd       3755 non-null   int64 
 7   employee_residence  3755 non-null   object
 8   remote_ratio        3755 non-null   int64 
 9   company_location    3755 non-null   object
 10  company_size        3755 non-null   object
dtypes: int64(4), object(7)
memory usage: 322.8+ KB
```

```
data.describe()
```

	work_year	salary	salary_in_usd	remote_ratio
count	3755.000000	3.755000e+03	3755.000000	3755.000000
mean	2022.373635	1.906956e+05	137570.389880	46.271638
std	0.691448	6.716765e+05	63055.625278	48.589050
min	2020.000000	6.000000e+03	5132.000000	0.000000
25%	2022.000000	1.000000e+05	95000.000000	0.000000
50%	2022.000000	1.380000e+05	135000.000000	0.000000
75%	2023.000000	1.800000e+05	175000.000000	100.000000
max	2023.000000	3.040000e+07	450000.000000	100.000000

```
data.isnull()
```

	work_year	experience_level	employment_type	job_title	salary	salary_cu
0	False	False	False	False	False	
1	False	False	False	False	False	
2	False	False	False	False	False	
3	False	False	False	False	False	
4	False	False	False	False	False	

```
result=data.dropna()
```

```
print(result)
```

```
      work_year experience_level employment_type                job_title  \
0          2023               SE              FT  Principal Data Scientist
1          2023               MI              CT               ML Engineer
2          2023               MI              CT               ML Engineer
3          2023               SE              FT            Data Scientist
4          2023               SE              FT            Data Scientist
...         ...              ...             ...                       ...
3750       2020               SE              FT            Data Scientist
3751       2021               MI              FT  Principal Data Scientist
3752       2020               EN              FT            Data Scientist
3753       2020               EN              CT     Business Data Analyst
3754       2021               SE              FT      Data Science Manager

       salary salary_currency  salary_in_usd employee_residence  remote_ratio
0       80000             EUR          85847                 ES           100
1       30000             USD          30000                 US           100
2       25500             USD          25500                 US           100
3      175000             USD         175000                 CA           100
4      120000             USD         120000                 CA           100
...       ...             ...            ...                ...           ...
3750   412000             USD         412000                 US           100
3751   151000             USD         151000                 US           100
3752   105000             USD         105000                 US           100
3753   100000             USD         100000                 US           100
3754  7000000             INR          94665                 IN            50

     company_location company_size
0                  ES            L
1                  US            S
2                  US            S
3                  CA            M
4                  CA            M
...               ...          ...
3750               US            L
3751               US            L
3752               US            S
3753               US            L
3754               IN            L
```

```
      [3755 rows x 11 columns]
```

```
count=data.isna().sum()
```

```
print(count)
```

```
      work_year             0
      experience_level      0
      employment_type       0
      job_title             0
      salary                0
      salary_currency       0
      salary_in_usd         0
      employee_residence    0
      remote_ratio          0
      company_location      0
      company_size          0
      dtype: int64
```

```
data.nunique()
```

```
      work_year                4
      experience_level         4
      employment_type          4
      job_title               93
      salary                 815
      salary_currency         20
      salary_in_usd         1035
      employee_residence      78
      remote_ratio             3
      company_location        72
      company_size             3
      dtype: int64
```

```
plt.scatter( data['experience_level'] ,data['salary'] )
plt.xlabel('experience_level')
plt.ylabel('salary')
plt.show()
```

```
X = data.drop('salary',axis=1)
y = data['salary']
```

```
X.shape , y.shape
```

```
((3755, 10), (3755,))
```

```
from sklearn.model_selection import train_test_split
X_train , X_test , Y_train , Y_test = train_test_split(X,y,random_state=101,test_size=0.2)
X_train.shape , X_test.shape , Y_train.shape , Y_test.shape
```

```
((3004, 10), (751, 10), (3004,), (751,))
```

```
data.corr()
```

```
<ipython-input-73-c44ded798807>:1: FutureWarning: The default value of numeric_
  data.corr()
```

	work_year	salary	salary_in_usd	remote_ratio
work_year	1.000000	-0.094724	0.228290	-0.236430
salary	-0.094724	1.000000	-0.023676	0.028731
salary_in_usd	0.228290	-0.023676	1.000000	-0.064171
remote_ratio	-0.236430	0.028731	-0.064171	1.000000

```
data1=data.drop(["work_year","employee_residence"],axis="columns")
```

```
data1.head(3)
```

	experience_level	employment_type	job_title	salary	salary_currency	salary_
0	SE	FT	Principal Data Scientist	80000	EUR	
1	MI	CT	ML Engineer	30000	USD	
2	MI	CT	ML Engineer	25500	USD	

```
data1.experience_level.unique()
```

```
array(['SE', 'MI', 'EN', 'EX'], dtype=object)
```

```
data1.salary_currency.unique()
```

```
array(['EUR', 'USD', 'INR', 'HKD', 'CHF', 'GBP', 'AUD', 'SGD', 'CAD',
       'ILS', 'BRL', 'THB', 'PLN', 'HUF', 'CZK', 'DKK', 'JPY', 'MXN',
       'TRY', 'CLP'], dtype=object)
```

```
data1.company_location.unique()
```

```
array(['ES', 'US', 'CA', 'DE', 'GB', 'NG', 'IN', 'HK', 'NL', 'CH', 'CF',
       'FR', 'FI', 'UA', 'IE', 'IL', 'GH', 'CO', 'SG', 'AU', 'SE', 'SI',
       'MX', 'BR', 'PT', 'RU', 'TH', 'HR', 'VN', 'EE', 'AM', 'BA', 'KE',
       'GR', 'MK', 'LV', 'RO', 'PK', 'IT', 'MA', 'PL', 'AL', 'AR', 'LT',
       'AS', 'CR', 'IR', 'BS', 'HU', 'AT', 'SK', 'CZ', 'TR', 'PR', 'DK',
       'BO', 'PH', 'BE', 'ID', 'EG', 'AE', 'LU', 'MY', 'HN', 'JP', 'DZ',
       'IQ', 'CN', 'NZ', 'CL', 'MD', 'MT'], dtype=object)
```

```
data1.company_size.unique()
```

```
array(['L', 'S', 'M'], dtype=object)
```

```
from sklearn.preprocessing import LabelEncoder
```

```
encoded_cols = ["experience_level","employment_type","salary_currency","company_location",
```

```
experience_level = LabelEncoder()
employment_type = LabelEncoder()
salary_currency = LabelEncoder()
company_location = LabelEncoder()
company_size = LabelEncoder()
```

```
data1['experience_levelEn'] = experience_level.fit_transform(data1["experience_level"])
data1["employment_typeEn"] = employment_type.fit_transform(data1["employment_type"])
data1["salary_currencyEn"] = salary_currency.fit_transform(data1["salary_currency"])
data1["company_locationEn"] = company_location.fit_transform(data1["company_location"])
data1["company_sizeEn"] = company_size.fit_transform(data1["company_size"])
```

```
data1.head(3)
```

	experience_level	employment_type	job_title	salary	salary_currency	salary_
0	SE	FT	Principal Data Scientist	80000	EUR	
1	MI	CT	ML Engineer	30000	USD	
2	MI	CT	ML Engineer	25500	USD	

```
data2 = data1.drop(encoded_cols,axis = "columns")
```

```
data2.head(3)
```

	salary	salary_in_usd	remote_ratio	experience_levelEn	employment_typeEn	sa
0	80000	85847	100	3	2	
1	30000	30000	100	2	0	
2	25500	25500	100	2	0	

```
jobs = data1.job_title.unique()
len(jobs)
```

```
93
```

```
jobs
```

```
array(['Principal Data Scientist', 'ML Engineer', 'Data Scientist',
       'Applied Scientist', 'Data Analyst', 'Data Modeler',
       'Research Engineer', 'Analytics Engineer',
       'Business Intelligence Engineer', 'Machine Learning Engineer',
       'Data Strategist', 'Data Engineer', 'Computer Vision Engineer',
       'Data Quality Analyst', 'Compliance Data Analyst',
       'Data Architect', 'Applied Machine Learning Engineer',
       'AI Developer', 'Research Scientist', 'Data Analytics Manager',
       'Business Data Analyst', 'Applied Data Scientist',
       'Staff Data Analyst', 'ETL Engineer', 'Data DevOps Engineer',
       'Head of Data', 'Data Science Manager', 'Data Manager',
       'Machine Learning Researcher', 'Big Data Engineer',
       'Data Specialist', 'Lead Data Analyst', 'BI Data Engineer',
       'Director of Data Science', 'Machine Learning Scientist',
       'MLOps Engineer', 'AI Scientist', 'Autonomous Vehicle Technician',
       'Applied Machine Learning Scientist', 'Lead Data Scientist',
       'Cloud Database Engineer', 'Financial Data Analyst',
       'Data Infrastructure Engineer', 'Software Data Engineer',
       'AI Programmer', 'Data Operations Engineer', 'BI Developer',
       'Data Science Lead', 'Deep Learning Researcher', 'BI Analyst',
       'Data Science Consultant', 'Data Analytics Specialist',
       'Machine Learning Infrastructure Engineer', 'BI Data Analyst',
       'Head of Data Science', 'Insight Analyst',
       'Deep Learning Engineer', 'Machine Learning Software Engineer',
       'Big Data Architect', 'Product Data Analyst',
       'Computer Vision Software Engineer', 'Azure Data Engineer',
       'Marketing Data Engineer', 'Data Analytics Lead', 'Data Lead',
       'Data Science Engineer', 'Machine Learning Research Engineer',
       'NLP Engineer', 'Manager Data Management',
       'Machine Learning Developer', '3D Computer Vision Researcher',
       'Principal Machine Learning Engineer', 'Data Analytics Engineer',
       'Data Analytics Consultant', 'Data Management Specialist',
       'Data Science Tech Lead', 'Data Scientist Lead',
```

```
            'Cloud Data Engineer', 'Data Operations Analyst',
            'Marketing Data Analyst', 'Power BI Developer',
            'Product Data Scientist', 'Principal Data Architect',
            'Machine Learning Manager', 'Lead Machine Learning Engineer',
            'ETL Developer', 'Cloud Data Architect', 'Lead Data Engineer',
            'Head of Machine Learning', 'Principal Data Analyst',
            'Principal Data Engineer', 'Staff Data Scientist',
            'Finance Data Analyst'], dtype=object)
```

```
job_title = LabelEncoder()
```

```
data3 = data1.drop(["job_title"],axis = "columns")
```

```
data3.head(3)
```

	experience_level	employment_type	salary	salary_currency	salary_in_usd	re
0	SE	FT	80000	EUR	85847	
1	MI	CT	30000	USD	30000	
2	MI	CT	25500	USD	25500	

```
data4 = data3.copy()
```

```
data4["salary_in_rupees"] = data3["salary_in_usd"].apply(lambda x: 79.82*x)
data4 = data4.drop(["salary_in_usd","salary"],axis=1)
```

```
data4.head()
```

	experience_level	employment_type	salary_currency	remote_ratio	company_loc
0	SE	FT	EUR	100	
1	MI	CT	USD	100	
2	MI	CT	USD	100	
3	SE	FT	USD	100	
4	SE	FT	USD	100	

```
sns.histplot(data['salary'])
```

```
<Axes: xlabel='salary', ylabel='Count'>
```


```
correlation =  data.corr()
```

```
<ipython-input-100-368159b823bb>:1: FutureWarning: The default value of numeric
  correlation =  data.corr()
```

```
x = data['work_year'].values
y = data['salary'].values
```

```
x_train, x_test, y_train, y_test = train_test_split(x,y,train_size = 0.8, test_size=0.2, r
x_train = x_train.reshape(-1,1)
x_test = x_test.reshape(-1,1)
x_train
```

```
array([[2022],
       [2023],
       [2021],
       ...,
       [2022],
       [2023],
       [2022]])
```

```
from sklearn.linear_model import LinearRegression
model = LinearRegression()
model.fit(x_train,y_train)
```

```
▾ LinearRegression
LinearRegression()
```

```
#prediction of output by the regression model
```

```
y_predict = model.predict(x_test.reshape(-1,1))
```

```
#model accuracy
```

```
train_accuracy = model.score(x_train,y_train)
train_accuracy
```

```
0.010197657178499187
```

```
test_accuracy = model.score(x_test,y_test)
test_accuracy
```

```
0.004627610408849625
```

```
#visual prediction
```

```
from sklearn.metrics import mean_squared_error
```

```
RMSE_model = mean_squared_error(y_test,y_predict,squared=False)
RMSE_model
```

```
1251516.1107562396
```

```
from sklearn.model_selection import cross_val_score
```

```
cross_val_score(model,x_train,y_train,cv=10)
```

```
array([-6.18773857e-03,  1.50777967e-03,  1.43354957e-02, -2.69227731e-02,
        4.29723548e-02, -7.45137626e-03,  9.65756458e-04,  3.56308708e-03,
        8.47066290e-03, -1.01065554e+00])
```

```
#random forest regression
```

```
from sklearn.ensemble import RandomForestRegressor
rf=RandomForestRegressor()
rf.fit(x_train,y_train)
```

```
▾ RandomForestRegressor
RandomForestRegressor()
```

```
rf.score(x_train,y_train)
```

```
0.02567118291893644
```

```
rf.score(x_test,y_test)
```

```
0.013008312320181625
```

```
from sklearn.tree import DecisionTreeRegressor
```

```
dt = DecisionTreeRegressor(random_state=0)
```

```
dt.fit(x_train,y_train)
```

```
▾ DecisionTreeRegressor
DecisionTreeRegressor(random_state=0)
```

```
y_predict = dt.predict(x_test)
```

```
RMSE_dt = mean_squared_error(y_test,y_predict,squared=False)
RMSE_dt
```

```
1246550.7137048095
```

```
cross_val_score(dt,x_train,y_train,cv=10)
```

```
array([-0.01237927,  0.04960442,  0.04736876, -0.01817865,  0.03695997,
       -0.04709274, -0.06126742,  0.02071357,  0.02916525, -1.32148158])
```

```
dt.score(x_test,y_test)
```

```
0.012510233011023608
```

```
dt.score(x_train,y_train)
```

```
0.025704167000316414
```

```
#logistic regression
```

```
from sklearn.linear_model import LogisticRegression
log_reg = LogisticRegression()
log_reg.fit(x_train,y_train)
```

```
▾ LogisticRegression
LogisticRegression()
```

```
y_predict_log= log_reg.predict(x_test)
```

```
from sklearn.metrics import accuracy_score
score_log = accuracy_score(y_test,y_predict_log)
print('Accuracy of model is : ',score_log)
```

```
Accuracy of model is :  0.03861517976031957
```

```
#KNN classification
```

```
from sklearn.neighbors import KNeighborsClassifier
knn = KNeighborsClassifier(n_neighbors=17)
knn.fit(x_train,y_train)
knn
```

```
KNeighborsClassifier
KNeighborsClassifier(n_neighbors=17)
```

```
y_predict_knn = knn.predict(x_test)
```

```
from sklearn.metrics import accuracy_score
score_knn = accuracy_score(y_test,y_predict_knn)
print('Accuracy of model is : ',score_knn)
```

```
Accuracy of model is :  0.0039946737683089215
```

```
#Support Vector Machine Algorithm(SVM algorithm)
```

```
from sklearn import svm
#from sklearn.svm import SVC
cv_classification = svm.SVC(kernel='rbf')
cv_classification.fit(x_train,y_train)
```

```
SVC
SVC()
```

```
y_predict_svm = cv_classification.predict(x_test)
```

```
from sklearn.metrics import accuracy_score
score_svm = accuracy_score(y_test,y_predict_svm)
print('Accuracy of model is : ',score_svm)
```

```
Accuracy of model is :  0.02663115845539281
```

```
#conclusion:
#logistic regression is efficient algorithm for this data salary prediction
```