Assignment

Linear Regression: Real Estate Training Dataset

Prepare four Models where Price as Dependent Variable with each other sqft_living, Bedroom, Bathroom, floors as Independent Variables.

Model 1: Price as DV & sqft_living as IDV

Step 1: Load the dataset

import pandas as pd

import matplotlib.pyplot as plt

import numpy as np

lr_realestate = pd.read_excel("D:/AI_ML_Course/Day21/Linear Regression.xlsx")

lr_realestate.columns

Out[28]: Index(['price', 'sqft_living', 'bedrooms', 'bathrooms', 'floors'], dtype='object')

import sklearn

y = lr_realestate.price

X = lr_realestate[['sqft_living']]

Step 2: Split the records for training & testing

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 = 2)

X_train.shape

Out[35]: (17290, 1)

X_test.shape

Out[36]: (4323, 1)

```
y_train.shape
```

Out[37]: (17290,)

```
y_test.shape
```

Out[38]: (4323,)

Step 3: Model Building with sklearn

```
from sklearn.linear_model import LinearRegression

lin_reg = LinearRegression()
```

Step 4: Train the model

```
lin_reg.fit(X_train, y_train)

lin_reg.coef_
```

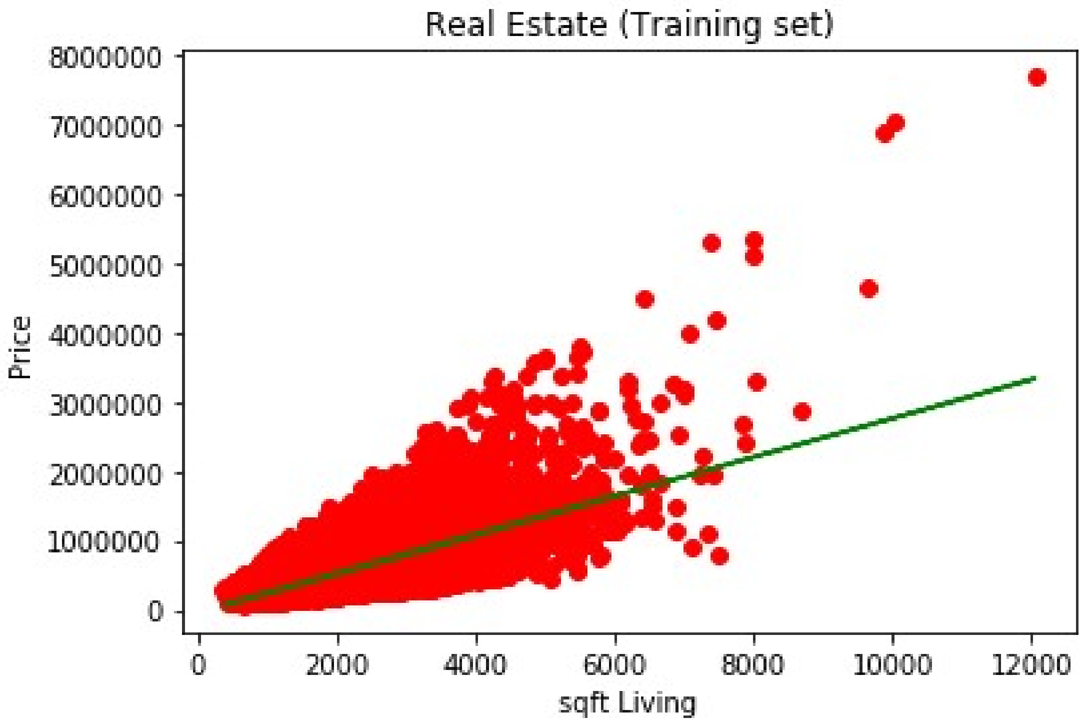
Out[42]: array([280.67382569])

```
lin_reg.intercept_
```

Out[43]: -42568.70358496299

Step 5: Visualize Training set result

```
plt.scatter(X_train, y_train, color = 'red')

plt.plot(X_train, lin_reg.predict(X_train), color = 'green')

plt.title('Real Estate (Training set)')

plt.xlabel('sqft_living')

plt.ylabel('Price')

plt.show()
```



Step 6: Test the Model

ypred=lin_reg.predict(X_test)

print(ypred)

array([633855.21632509, 566493.49815977, 364408.34366382, ...,

701216.9344904 , 187583.83347987, 465450.9209118])

X_test.head()

Out[111]:

	sqft_living
6638	2410
7366	2170
3158	1450
9117	4500
3392	860

Step 7: Estimate the cost

```
from sklearn.metrics import mean_squared_error,r2_score

RMSE=np.sqrt(mean_squared_error(y_test,ypred))

r_square=r2_score(y_test,ypred)

print('The R-Square value is...',r_square)
```

The R-Square value is... 0.5031163723285275

```
print('The RMSE value is........',RMSE)
```

The RMSE value is........ 263380.00189817196

Step 8: How to predict for unseen value

```
unseen_pred=lin_reg.predict(np.array([[2410]]))

print('The unseen for the given x is....',unseen_pred)
```

The unseen for the given x is.... [633855.21632509]

Model 2: Price as DV & Bedroom as IDV

Step 1: Load the dataset

```
import pandas as pd

import matplotlib.pyplot as plt

import numpy as np

lr_realestate = pd.read_excel("D:/AI_ML_Course/Day21/Linear Regression.xlsx")

lr_realestate.columns

Out[28]: Index(['price', 'sqft_living', 'bedrooms', 'bathrooms', 'floors'], dtype='object')


import sklearn

y = lr_realestate.price

X = lr_realestate[['bedrooms’]]
```

Step 2: Split the records for training & testing

```
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 = 2)

X_train.shape

Out[35]: (17290, 1)

X_test.shape

Out[36]: (4323, 1)

y_train.shape

Out[37]: (17290,)

y_test.shape

Out[38]: (4323,)
```

Step 3: Model Building with sklearn

```
from sklearn.linear_model import LinearRegression

lin_reg = LinearRegression()
```

Step 4: Train the model

```
lin_reg.fit(X_train, y_train)

lin_reg.coef_
```

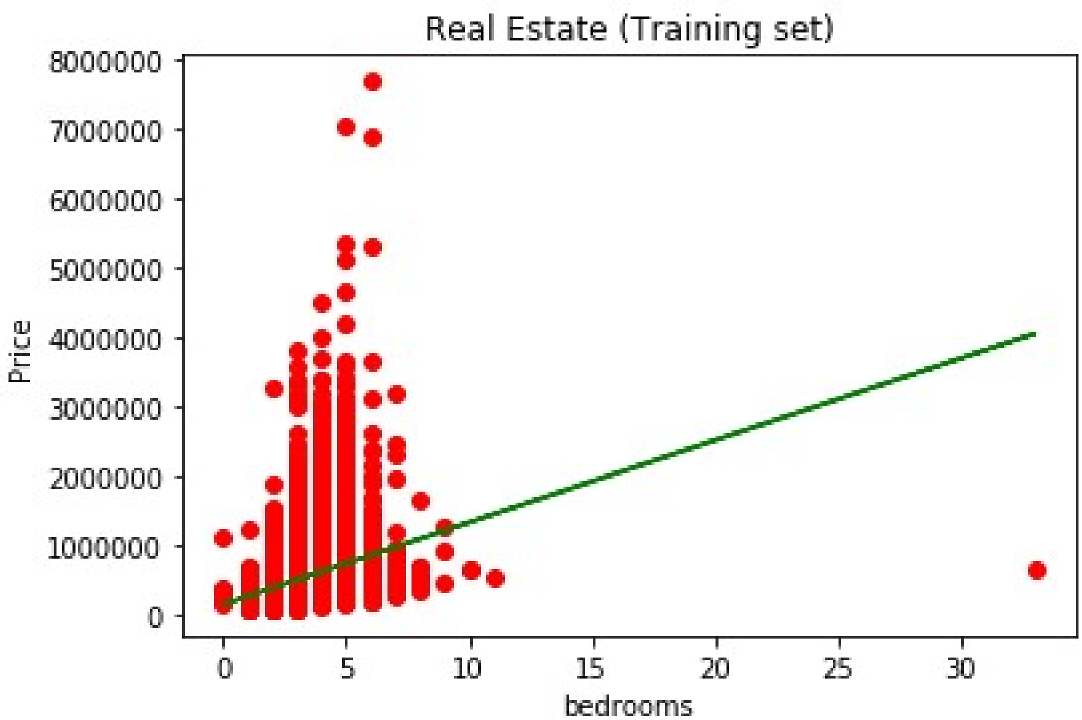
```
 array([118660.62797869])
```

```
lin_reg.intercept_
```

```
139952.8759338616
```

Step 5: Visualize Training set result

```
plt.scatter(X_train, y_train, color = 'red')

plt.plot(X_train, lin_reg.predict(X_train), color = 'green')

plt.title('Real Estate (Training set)')

plt.xlabel('sqft_living')

plt.ylabel('Price')

plt.show()
```



Step 6: Test the Model

ypred=lin_reg.predict(X_test)

print(ypred)

array([614595.3878486 , 495934.75986992, 377274.13189123, ...,

614595.3878486 , 377274.13189123, 614595.3878486])

X_test.head()

Out[135]:

	bedrooms
6638	4
7366	3
3158	2
9117	5
3392	2

Step 7: Estimate the cost

```
from sklearn.metrics import mean_squared_error,r2_score

RMSE=np.sqrt(mean_squared_error(y_test,ypred))

r_square=r2_score(y_test,ypred)

print('The R-Square value is...',r_square)
```

```
The R-Square value is... 0.10886345250291585
```

```
print('The RMSE value is........',RMSE)
```

```
The RMSE value is........ 352717.9654187645
```

Step 8: How to predict for unseen value

```
unseen_pred=lin_reg.predict(np.array([[3]]))

print('The unseen for the given x is....',unseen_pred)
```

```
The unseen for the given x is.... [377274.13189123]
```

Model 3: Price as DV & Bathroom as IDV

Step 1: Load the dataset

```
import pandas as pd

import matplotlib.pyplot as plt

import numpy as np

lr_realestate = pd.read_excel("D:/AI_ML_Course/Day21/Linear Regression.xlsx")

lr_realestate.columns

Out[28]: Index(['price', 'sqft_living', 'bedrooms', 'bathrooms', 'floors'], dtype='object')


import sklearn

y = lr_realestate.price

X = lr_realestate[['bathrooms’]]
```

Step 2: Split the records for training & testing

```
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 = 2)

X_train.shape

Out[35]: (17290, 1)

X_test.shape

Out[36]: (4323, 1)

y_train.shape

Out[37]: (17290,)

y_test.shape

Out[38]: (4323,)
```

Step 3: Model Building with sklearn

```
from sklearn.linear_model import LinearRegression

lin_reg = LinearRegression()
```

Step 4: Train the model

```
lin_reg.fit(X_train, y_train)

lin_reg.coef_
```

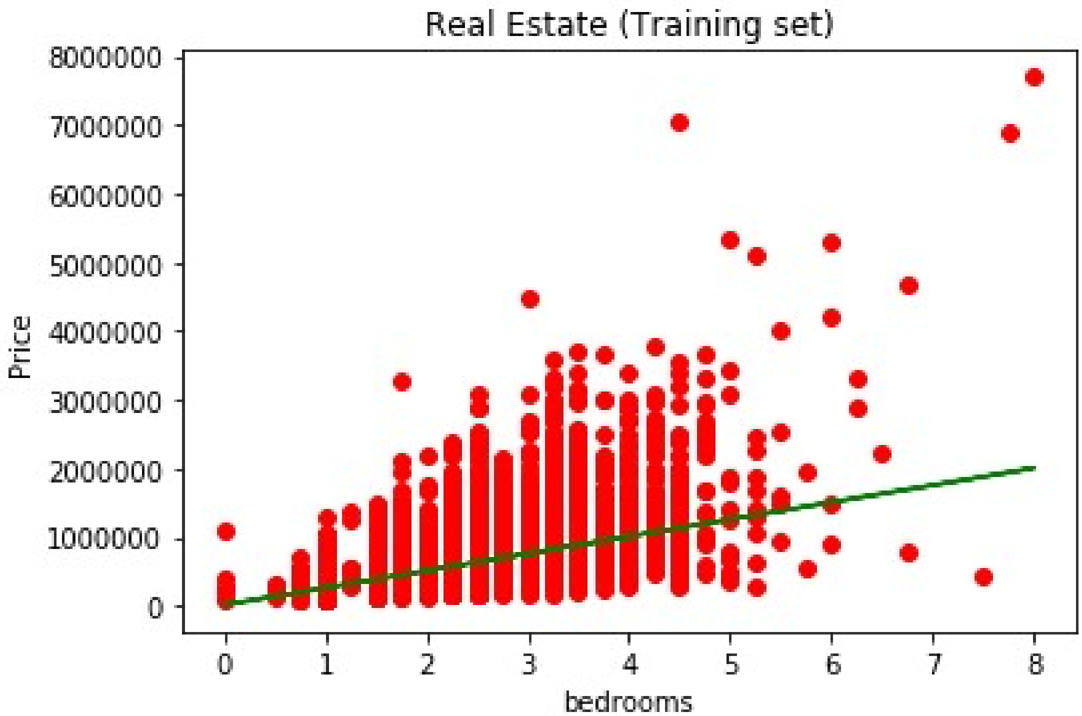
```
array([249143.95803858])
```

```
lin_reg.intercept_
```

```
13073.99575
```

Step 5: Visualize Training set result

```
plt.scatter(X_train, y_train, color = 'red')

plt.plot(X_train, lin_reg.predict(X_train), color = 'green')

plt.title('Real Estate (Training set)')

plt.xlabel('sqft_living')

plt.ylabel('Price')

plt.show()
```



Step 6: Test the Model

ypred=lin_reg.predict(X_test)

print(ypred)

array([573647.90133969, 386789.93281076, 262217.95379147, ...,

635933.89084934, 262217.95379147, 698219.88035898])

X_test.head()

Out[170]:

	bathrooms
6638	2.25
7366	1.50
3158	1.00
9117	3.25
3392	1.00

Step 7: Estimate the cost

```
from sklearn.metrics import mean_squared_error,r2_score

RMSE=np.sqrt(mean_squared_error(y_test,ypred))

r_square=r2_score(y_test,ypred)

print('The R-Square value is...',r_square)
```

The R-Square value is... 0.28122887124177365

```
print('The RMSE value is........',RMSE)
```

The RMSE value is........ 316774.90190998075

Step 8: How to predict for unseen value

```
unseen_pred=lin_reg.predict(np.array([[2.25]]))

print('The unseen for the given x is....',unseen_pred)
```

The unseen for the given x is.... [573647.90133969]

Model 4: Price as DV & Floors as IDV

Step 1: Load the dataset

```
import pandas as pd

import matplotlib.pyplot as plt

import numpy as np

lr_realestate = pd.read_excel("D:/AI_ML_Course/Day21/Linear Regression.xlsx")

lr_realestate.columns

Out[28]: Index(['price', 'sqft_living', 'bedrooms', 'bathrooms', 'floors'], dtype='object')


import sklearn

y = lr_realestate.price

X = lr_realestate[['floors’]]
```

Step 2: Split the records for training & testing

```
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 = 2)

X_train.shape

Out[35]: (17290, 1)

X_test.shape

Out[36]: (4323, 1)

y_train.shape

Out[37]: (17290,)

y_test.shape

Out[38]: (4323,)
```

Step 3: Model Building with sklearn

```
from sklearn.linear_model import LinearRegression

lin_reg = LinearRegression()
```

Step 4: Train the model

```
lin_reg.fit(X_train, y_train)

lin_reg.coef_
```

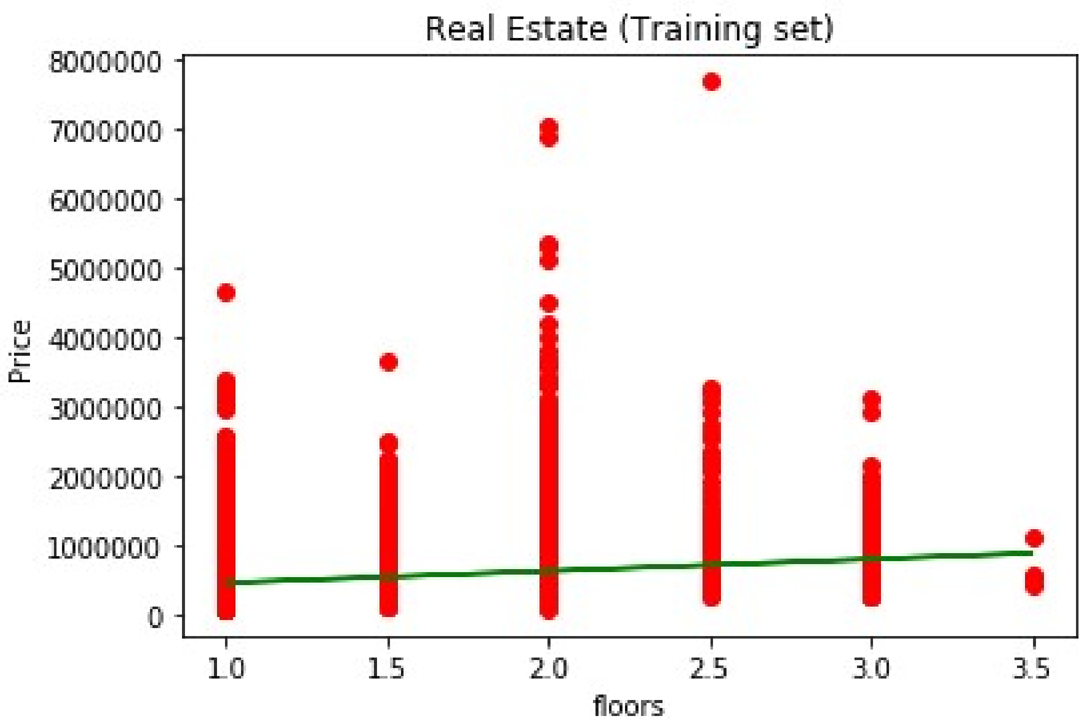
```
array([171376.44562902])
```

```
lin_reg.intercept_
```

```
283309.93245028483
```

Step 5: Visualize Training set result

```
plt.scatter(X_train, y_train, color = 'red')

plt.plot(X_train, lin_reg.predict(X_train), color = 'green')

plt.title('Real Estate (Training set)')

plt.xlabel('floors')

plt.ylabel('Price')

plt.show()
```



Step 6: Test the Model

ypred=lin_reg.predict(X_test)

print(ypred)

array([540374.60089382, 454686.37807931, 454686.37807931, ...,

626062.82370833, 454686.37807931, 540374.60089382])

X_test.head()

Out[197]:

	floors
6638	1.5
7366	1.0
3158	1.0
9117	2.0
3392	1.0

Step 7: Estimate the cost

```
from sklearn.metrics import mean_squared_error,r2_score

RMSE=np.sqrt(mean_squared_error(y_test,ypred))

r_square=r2_score(y_test,ypred)

print('The R-Square value is...',r_square)
```

The R-Square value is... 0.0733487976687478

```
print('The RMSE value is........',RMSE)
```

The RMSE value is........ 359677.77234107786

Step 8: How to predict for unseen value

```
unseen_pred=lin_reg.predict(np.array([[2.25]]))

print('The unseen for the given x is....',unseen_pred)
```

The unseen for the given x is.... [626062.82370833]

Logistic Regression

Project 1: Bank Loan Modeling

Dependent Variable as “Personal Loan” and Independent Variable as “others”

Step 1: Load the Dataset

```
import pandas as pd

bankPL_dataset =
pd.read_excel("D:/AI_ML_Course/Day24/Bank_Personal_Loan_Modelling.xlsx",
sheet_name='Data')

bankPL_dataset.columns

# Index(['ID', 'Age', 'Experience', 'Income', 'ZIP Code', 'Family', 'CCAvg',

#        'Education', 'Mortgage', 'Personal Loan', 'Securities Account',

#        'CD Account', 'Online', 'CreditCard'],

#       dtype='object')

bankPL_dataset = bankPL_dataset.drop(['ID','ZIP Code'], axis=1)

bankPL_dataset.columns

# Index(['Age', 'Experience', 'Income', 'Family', 'CCAvg', 'Education',

#        'Mortgage', 'Personal Loan', 'Securities Account', 'CD Account',

#        'Online', 'CreditCard'],

#       dtype='object')
```

Step 2: Logistic Regression

```
import statsmodels.api as sm

Y = bankPL_dataset['Personal Loan']

X = bankPL_dataset[['Age', 'Experience', 'Income', 'Family', 'CCAvg', 'Education', 'Mortgage',
'Securities Account', 'CD Account', 'Online', 'CreditCard']]
```

```
X1 = sm.add_constant(X)

BanKPL = sm.Logit(Y, X1)

result = BanKPL.fit()

# Optimization terminated successfully.

#         Current function value: 0.128435

#         Iterations 9

result.summary()
```

Logit Regression Results

==

Dep. Variable:	Personal Loan	No. Observations:	5000
Model:	Logit	Df Residuals:	4988
Method:	MLE	Df Model:	11
Date:	Wed, 12 Aug 2020	Pseudo R-squ.:	0.5938
Time:	16:42:50	Log-Likelihood:	-642.18
converged:	True	LL-Null:	-1581.0
Covariance Type:	nonrobust	LLR p-value:	0.000

==

	coef	std err	z	P>|z|	[0.025	0.975]
const	-12.1928	1.645	-7.411	0.000	-15.417	-8.968
Age	-0.0536	0.061	-0.874	0.382	-0.174	0.067
Experience	0.0638	0.061	1.046	0.295	-0.056	0.183
Income	0.0546	0.003	20.831	0.000	0.049	0.060
Family	0.6958	0.074	9.364	0.000	0.550	0.841

CCAvg	0.1240	0.040	3.127	0.002	0.046	0.202
Education	1.7362	0.115	15.088	0.000	1.511	1.962
Mortgage	0.0005	0.001	0.856	0.392	-0.001	0.002
Securities Account	-0.9368	0.286	-3.277	0.001	-1.497	-0.377
CD Account	3.8225	0.324	11.800	0.000	3.188	4.457
Online	-0.6752	0.157	-4.298	0.000	-0.983	-0.367
CreditCard	-1.1197	0.205	-5.462	0.000	-1.522	-0.718

==

Step 3: Find the significant variables

With the above Logit Regression table, the variable where the P value is less than 0.05 is the significant variable.

So the significant variables are:

- Income
- Family
- CCAvg
- Education
- Securities Account
- CD Account
- Online
- CreditCard

Project 2: Attrition Rate Analysis

Dependent Variable as “Attrition” and Independent Variable as “others”

Step 1: Load the Dataset

```
import pandas as pd

attrition_dataset = pd.read_csv("D:/AI_ML_Course/Day24/general_data.csv")

attrition_dataset.columns

# Index(['Age', 'Attrition', 'BusinessTravel', 'Department', 'DistanceFromHome',
#        'Education', 'EducationField', 'EmployeeCount', 'EmployeeID', 'Gender',
#        'JobLevel', 'JobRole', 'MaritalStatus', 'MonthlyIncome',
#        'NumCompaniesWorked', 'Over18', 'PercentSalaryHike', 'StandardHours',
#        'StockOptionLevel', 'TotalWorkingYears', 'TrainingTimesLastYear',
#        'YearsAtCompany', 'YearsSinceLastPromotion', 'YearsWithCurrManager'],
#       dtype='object')


attrition_dataset = attrition_dataset.drop(['EmployeeCount',
'EmployeeID','Over18','StandardHours'], axis=1)

attrition_dataset.columns

# Index(['Age', 'Attrition', 'BusinessTravel', 'Department', 'DistanceFromHome',
#        'Education', 'EducationField', 'Gender', 'JobLevel', 'JobRole',
#        'MaritalStatus', 'MonthlyIncome', 'NumCompaniesWorked',
#        'PercentSalaryHike', 'StockOptionLevel', 'TotalWorkingYears',
#        'TrainingTimesLastYear', 'YearsAtCompany', 'YearsSinceLastPromotion',
#        'YearsWithCurrManager'],
#       dtype='object')
```

```
from sklearn import preprocessing

attrition_dataset = attrition_dataset.dropna()

attrition_dataset = attrition_dataset.drop_duplicates()

le = preprocessing.LabelEncoder()

attrition_dataset['Attrition'] = le.fit_transform(attrition_dataset['Attrition'])

attrition_dataset['BusinessTravel'] = le.fit_transform(attrition_dataset['BusinessTravel'])

attrition_dataset['Department'] = le.fit_transform(attrition_dataset['Department'])

attrition_dataset['EducationField'] = le.fit_transform(attrition_dataset['EducationField'])

attrition_dataset['Gender'] = le.fit_transform(attrition_dataset['Gender'])

attrition_dataset['JobRole'] = le.fit_transform(attrition_dataset['JobRole'])

attrition_dataset['MaritalStatus'] = le.fit_transform(attrition_dataset['MaritalStatus'])
```

Step 2: Logistic Regression

```
import statsmodels.api as sm

Y = attrition_dataset['Attrition']

X = attrition_dataset[['Age', 'BusinessTravel', 'Department', 'DistanceFromHome', 'Education',
'EducationField', 'Gender', 'JobLevel', 'JobRole', 'MaritalStatus', 'MonthlyIncome',
'NumCompaniesWorked', 'PercentSalaryHike', 'StockOptionLevel', 'TotalWorkingYears',
'TrainingTimesLastYear', 'YearsAtCompany', 'YearsSinceLastPromotion',
'YearsWithCurrManager']]

X1 = sm.add_constant(X)

Logistic_Attrition = sm.Logit(Y, X1)

result = Logistic_Attrition.fit()

# Optimization terminated successfully.

#         Current function value: 0.392756

#         Iterations 7
```

result.summary()

Logit Regression Results

```
==============================================================================
Dep. Variable:           Attrition           No. Observations:          1470
Model:                   Logit               Df Residuals:              1450
Method:                  MLE                 Df Model:                    19
Date:            Sun, 16 Aug 2020            Pseudo R-squ.:           0.1108
Time:                    23:20:53            Log-Likelihood:         -577.35
converged:               True                LL-Null:                -649.29
Covariance Type:         nonrobust           LLR p-value:          3.295e-21
==============================================================================
=============
                     coef    std err      z      P>|z|     [0.025     0.975]
-----------------------------------------------------------------------------
const               0.0650    0.717    0.091    0.928    -1.340    1.470
Age                -0.0306    0.012   -2.583    0.010    -0.054   -0.007
BusinessTravel     -0.0166    0.113   -0.146    0.884    -0.239    0.206
Department         -0.2421    0.141   -1.720    0.085    -0.518    0.034
DistanceFromHome   -0.0014    0.009   -0.145    0.884    -0.020    0.017
Education          -0.0625    0.074   -0.847    0.397    -0.207    0.082
EducationField     -0.0965    0.058   -1.669    0.095    -0.210    0.017
Gender              0.0869    0.155    0.560    0.576    -0.217    0.391
```

JobLevel	-0.0249	0.069	-0.363	0.717	-0.159	0.110
JobRole	0.0378	0.031	1.219	0.223	-0.023	0.099
MaritalStatus	0.5885	0.109	5.379	0.000	0.374	0.803
MonthlyIncome	-1.868e-06	1.66e-06	-1.128	0.259	-5.11e-06	1.38e-06
NumCompaniesWorked	0.1184	0.032	3.729	0.000	0.056	0.181
PercentSalaryHike	0.0117	0.020	0.576	0.565	-0.028	0.052
StockOptionLevel	-0.0645	0.089	-0.721	0.471	-0.240	0.111
TotalWorkingYears	-0.0593	0.021	-2.856	0.004	-0.100	-0.019
TrainingTimesLastYear	-0.1465	0.061	-2.406	0.016	-0.266	-0.027
YearsAtCompany	0.0136	0.032	0.428	0.669	-0.049	0.076
YearsSinceLastPromotion	0.1323	0.035	3.732	0.000	0.063	0.202
YearsWithCurrManager	-0.1396	0.038	-3.642	0.000	-0.215	-0.064

==

Step 3: Find the significant variables

With the above Logit Regression table, the variable where the P value is less than 0.05 is the significant variable.

So the significant variables are:

- Age
- Marital Status
- NumCompaniesWorked
- TotalWorkingYears
- TrainingTimeLastYear
- YearsSinceLastPromotion
- YearsWithCurrManager