DB Project Part 3

August 10, 2023

[1]:
```
import pandas as pd
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn import metrics
from sklearn.metrics import r2_score
from sklearn.model_selection import GridSearchCV
from sklearn.linear_model import Lasso, Ridge
from scipy import stats
import pickle
```

1 Load necessary tables

[2]:
```
%%bigquery product
SELECT * FROM full_insurance_data.ProductType
```

```
Query is running:   0%|          |
```

```
Downloading:   0%|          |
```

[3]:
```
%%bigquery Disease
SELECT
    HP.ID,
    HP.Health_Condition_ID,
    HP.Hypertension,
    HP.High_Cholesterol,
    HP.CoronaryHeartDisease,
    HP.Angina,
    HP.HeartAttack,
    HP.Stroke,
    HP.Asthma,
    HP.Cancer,
    HP.Prediabetes,
    HP.GestationalDiabetes,
    HP.COPD,
```

```
    HP.Arthritis,
    HP.Dementia,
    HP.Anxiety_Disorder,
    HP.Depression,
    HP.Epilepsy,
    CC.Chronic_Fatigue_Syndrome
FROM
    full_insurance_data.Health_Problem AS HP
JOIN
    full_insurance_data.Current_Conditions AS CC
ON
    HP.ID = CC.ID
```

Query is running: 0%| |

Downloading: 0%| |

[4]:
```
# List of disease columns to check
disease_columns = [
    'Hypertension', 'High_Cholesterol', 'CoronaryHeartDisease', 'Angina',
    'HeartAttack', 'Stroke', 'Asthma', 'Cancer', 'Prediabetes',
    'GestationalDiabetes', 'COPD', 'Arthritis', 'Dementia',
    'Anxiety_Disorder', 'Depression', 'Epilepsy', 'Chronic_Fatigue_Syndrome'
]

# Function to check if any of the diseases is equal to 1 for a row
def has_disease(row):
    return any(pd.notna(row[col]) and row[col] == 1 for col in disease_columns)

# Create the 'disease' variable based on the custom function
Disease['disease'] = Disease.apply(has_disease, axis=1)
```

[5]:
```
%%bigquery Current_Condition
SELECT
    ID,
    Health_Condition_ID,
    Weight,
    Height,
    Pregnant,
    Health_WeakImmune,
    (Weight * 0.45359237) / POWER((Height * 0.0254), 2) AS BMI
FROM
    full_insurance_data.Current_Conditions
```

Query is running: 0%| |

Downloading: 0%| |

[6]:
```
%%bigquery Lifestyle
SELECT *
FROM full_insurance_data.Alcohol AS A
JOIN full_insurance_data.smoking AS S
ON A.ID = S.ID
JOIN full_insurance_data.activity AS Act
ON A.ID = Act.ID
```

```
Query is running:   0%|          |

Downloading:   0%|          |
```

[7]:
```
%%bigquery demographics
SELECT ID,
 ↪Urban_Rural,Region,Gender,Age,Race,Education,Num_Fam_Adult,Num_Fam_Kid,
 ↪Current_MaritalStatus,Citizenship,JobYN,Housing
FROM full_insurance_data.Demographic
```

```
Query is running:   0%|          |

Downloading:   0%|          |
```

2 Preprocessing

[77]:
```
merged_df = demographics.merge(Current_Condition, on='ID').merge(Disease,
 ↪on='ID').merge(Lifestyle,on='ID').merge(product, on='ID')
merged_df
```

[77]:
```
            ID  Urban_Rural  Region  Gender  Age  Race  Education  \
0      H025402            4       4       1   35     8         97
1      H022119            1       4       2   38     7         97
2      H022203            1       2       2   27     8          5
3      H036978            1       1       1   54     1          8
4      H032270            1       4       2   44     1          5
…          …          …     …     …  …   …        …
27646  H030644            1       4       1   21     6          4
27647  H056560            3       2       2   20     6          4
27648  H052927            3       2       2   20     6          5
27649  H057497            2       3       2   29     6          1
27650  H011969            2       4       2   58     6          9

       Num_Fam_Adult  Num_Fam_Kid  Current_MaritalStatus  …  Walking  \
0                  2            2                      8  …        1
1                  2            2                      1  …        1
2                  1            0                      7  …        1
3                  1            0                      9  …        1
4                  3            1                      1  …        1
…              …            …                    …  …      …
```

```
27646                 1             0                         9  …          1
27647                 2             0                         9  …          1
27648                 3             0                         9  …          2
27649                 1             0                         9  …          1
27650                 2             0                         9  …          1

       Sleeping  Eating  Meditation  Yoga  Therapy  Dr_Visit  Coverage  \
0             9       1           2     2        2         0         1
1             5       1           2     2        2         0         1
2             6       4           2     2        2         1         2
3             6       1           2     7        2         0         2
4             8       1           1     1        1         0         2
…           …       …         …     …       …        …
27646        98       8           8     8        2         0         1
27647        98       8           8     8        2         0         2
27648        98       8           8     8        1         5         2
27649        98       8           8     8        8         8         2
27650        98       7           8     8        2         0         2

       ProductType  Premium
0                4     <NA>
1                4     <NA>
2                2     <NA>
3                1    99999
4                1     7200
…              …       …
27646            4     <NA>
27647            2     <NA>
27648            1    99999
27649            2     <NA>
27650            1     <NA>

[27651 rows x 55 columns]
```

[93]:
```
merged_df
```

[93]:
```
            ID  Urban_Rural  Region  Gender  Age  Race  Education  \
3      H036978            1       1       1   54     1          8
4      H032270            1       4       2   44     1          5
5      H058634            1       4       2   19     1          5
13     H012186            1       2       1   31     1          5
15     H006593            1       1       1   34     2          8
…          …          …     …       …   …    …        …
27643  H025641            3       2       1   42     6          3
27644  H065773            1       3       2   27     6          1
27645  H052100            1       3       2   41     6          8
27648  H052927            3       2       2   20     6          5
```

```
27650  H011969            2       4       2   58     6          9

       Num_Fam_Adult  Num_Fam_Kid  Current_MaritalStatus  …  Eating  \
3                  1            0                      9  …       1
4                  3            1                      1  …       1
5                  2            1                      7  …       2
13                 2            0                      9  …       1
15                 2            3                      9  …       1
…                …          …                    …  …     …
27643              2            0                      9  …       8
27644              2            3                      1  …       1
27645              3            2                      9  …       8
27648              3            0                      9  …       8
27650              2            0                      9  …       7

       Meditation  Yoga  Therapy  Dr_Visit  Coverage  ProductType  Premium  \
3               2     7        2         0         2            1    99999
4               1     1        1         0         2            1     7200
5               2     2        2         0         2            1    99999
13              2     2        2         0         2            1    99999
15              2     2        2         0         2            1     4800
…             …   …      …       …       …         …
27643           8     8        8         8         2            1    99999
27644           8     8        2         0         2            1      660
27645           8     8        8         8         2            1     4392
27648           8     8        1         5         2            1    99999
27650           8     8        2         0         2            1     4800

       LogPremium  smoke
3       11.512915      0
4        8.881836      0
5       11.512915      0
13      11.512915      0
15       8.476371      0
…             …    …
27643   11.512915      2
27644     6.49224      0
27645     8.38754      2
27648   11.512915      2
27650    8.476371      2

[12928 rows x 57 columns]
```

[79]:
```python
# Drop people without insurance
merged_df = merged_df[merged_df['ProductType'] == 1]

# Replace NaN values in 'Premium' with the median value
```

```python
median_value = merged_df['Premium'].median()
merged_df['Premium'] = merged_df['Premium'].fillna(median_value)
#add log premium
merged_df['LogPremium']=np.log(merged_df['Premium'])
#Add smoke yes/no variable - 1=Yes, 0=No, 2=Unknown
smoke_mapping={1:1,2:1, 3:0,4:0,5:2,9:2}
merged_df['smoke'] = merged_df['SMKCIGST_A'].map(smoke_mapping)

#merged_df

#check missing data
#remove those with NA premiums
merged_df['Chronic_Fatigue_Syndrome']=merged_df['Chronic_Fatigue_Syndrome'].
↪fillna(8)
merged_df.loc[merged_df.Gender==1&merged_df.Pregnant,'Pregnant']=2
merged_df['Pregnant']=merged_df['Pregnant'].fillna(8)

merged_df.isnull().sum()
```

```
/var/tmp/ipykernel_8343/3600034072.py:6: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  merged_df['Premium'] = merged_df['Premium'].fillna(median_value)
/var/tmp/ipykernel_8343/3600034072.py:8: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  merged_df['LogPremium']=np.log(merged_df['Premium'])
/var/tmp/ipykernel_8343/3600034072.py:11: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  merged_df['smoke'] = merged_df['SMKCIGST_A'].map(smoke_mapping)
/var/tmp/ipykernel_8343/3600034072.py:17: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  merged_df['Chronic_Fatigue_Syndrome']=merged_df['Chronic_Fatigue_Syndrome'].fi
llna(8)
```

```
/var/tmp/ipykernel_8343/3600034072.py:19: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  merged_df['Pregnant']=merged_df['Pregnant'].fillna(8)
```

[79]:
```
ID                            0
Urban_Rural                   0
Region                        0
Gender                        0
Age                           0
Race                          0
Education                     0
Num_Fam_Adult                 0
Num_Fam_Kid                   0
Current_MaritalStatus         0
Citizenship                   0
JobYN                      2243
Housing                       0
Health_Condition_ID_x         0
Weight                        0
Height                        0
Pregnant                      0
Health_WeakImmune             0
BMI                           0
Health_Condition_ID_y         0
Hypertension                  0
High_Cholesterol              0
CoronaryHeartDisease          0
Angina                        0
HeartAttack                   0
Stroke                        0
Asthma                        0
Cancer                        0
Prediabetes                   0
GestationalDiabetes           0
COPD                          0
Arthritis                     0
Dementia                      0
Anxiety_Disorder              0
Depression                    0
Epilepsy                      0
Chronic_Fatigue_Syndrome      0
disease                       0
Life_Style_ID                 0
```

```
DRKSTAT_A                  0
ID_1                       0
Life_Style_ID_1            0
SMKCIGST_A                 0
ID_2                       0
Life_Style_ID_2            0
Walking                    0
Sleeping                   0
Eating                     0
Meditation                 0
Yoga                       0
Therapy                    0
Dr_Visit                   0
Coverage                   0
ProductType                0
Premium                    0
LogPremium                 0
smoke                      0
dtype: int64
```

3 Linear Regression - Prediction of Insurance Premium Depending of different variables

[80]:

```
# dividing dataset into train and test
x = merged_df[['Gender', 'Age',␣
 ↪'BMI','Num_Fam_Adult','Health_WeakImmune','Num_Fam_Kid','SMKCIGST_A','DRKSTAT_A',
              'Hypertension', 'High_Cholesterol', 'CoronaryHeartDisease',␣
 ↪'Angina',
   'HeartAttack', 'Stroke', 'Asthma', 'Cancer',
   'GestationalDiabetes', 'COPD', 'Arthritis', 'Dementia',
   'Anxiety_Disorder', 'Depression', 'Epilepsy',␣
 ↪'Chronic_Fatigue_Syndrome','Citizenship','Urban_Rural','Education']]
y = merged_df[['LogPremium']]

# Split 20% with test_size=0.2
X_train, X_test, y_train, y_test = train_test_split(x, y, test_size=0.2,␣
 ↪random_state=42)
print(x.shape, y.shape)
print(X_train.shape, y_train.shape)
print(X_test.shape, y_test.shape)
```

```
(12928, 27) (12928, 1)
(10342, 27) (10342, 1)
(2586, 27) (2586, 1)
```

[38]:
```
cat_cols = [
    'Hypertension', 'High_Cholesterol', 'CoronaryHeartDisease', 'Angina',
    'HeartAttack', 'Stroke', 'Asthma', 'Cancer',
    'GestationalDiabetes', 'COPD', 'Arthritis', 'Dementia',
    'Anxiety_Disorder', 'Depression', 'Epilepsy',␣
 ↪'Chronic_Fatigue_Syndrome','Citizenship','Urban_Rural',
'Education']
for c in cat_cols:
    X_train[c]=X_train[c].astype("category")
    X_test[c]=X_test[c].astype("category")
```

[97]:
```
len(X_train.columns)
```

[97]: 27

[81]:
```
#Linear regression
model = LinearRegression()
model.fit(X_train, y_train)
train_pred = model.predict(X_train)

# calculate the accuracy of the model by computing the R2 score between␣
 ↪predicted and real values
r2_train = metrics.r2_score(y_train, train_pred)
spearman=print('R squared value : ', r2_train)
```

```
R squared value :  0.019363748735138686
```

[82]:
```
# prediction on test data
test_pred =model.predict(X_test)
res = stats.spearmanr(y_test, test_pred)
# R squared value
r2_test = metrics.r2_score(y_test, test_pred)
print('R squared value : ', r2_test)
print('Spearman Rank : ', res.statistic)
```

```
R squared value :  0.017303552597084O5
Spearman Rank :  0.15299235196351244
```

[92]:
```
model.intercept_
```

[92]: array([7.47009931])

[90]:
```
X_train['Age'].dtype
```

[90]: Int64Dtype()

[83]:
```
import statsmodels.api as sm

X2 = sm.add_constant(X_train)
```

```python
est = sm.OLS(y_train, X2)
est2 = est.fit()
print(est2.summary())
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[83], line 4
      1 import statsmodels.api as sm
      3 X2 = sm.add_constant(X_train)
----> 4 est = sm.OLS(y_train, X2)
      5 est2 = est.fit()
      6 print(est2.summary())

File /opt/conda/lib/python3.10/site-packages/statsmodels/regression/linear_model.
↪py:922, in OLS.__init__(self, endog, exog, missing, hasconst, **kwargs)
    919     msg = ("Weights are not supported in OLS and will be ignored"
    920            "An exception will be raised in the next version.")
    921     warnings.warn(msg, ValueWarning)
--> 922 super(OLS, self).__init__(endog, exog, missing=missing,
    923                           hasconst=hasconst, **kwargs)
    924 if "weights" in self._init_keys:
    925     self._init_keys.remove("weights")

File /opt/conda/lib/python3.10/site-packages/statsmodels/regression/linear_model.
↪py:748, in WLS.__init__(self, endog, exog, weights, missing, hasconst,␣
↪**kwargs)
    746 else:
    747     weights = weights.squeeze()
--> 748 super(WLS, self).__init__(endog, exog, missing=missing,
    749                           weights=weights, hasconst=hasconst, **kwargs)
    750 nobs = self.exog.shape[0]
    751 weights = self.weights

File /opt/conda/lib/python3.10/site-packages/statsmodels/regression/linear_model.
↪py:202, in RegressionModel.__init__(self, endog, exog, **kwargs)
    201 def __init__(self, endog, exog, **kwargs):
--> 202     super(RegressionModel, self).__init__(endog, exog, **kwargs)
    203     self.pinv_wexog: Float64Array | None = None
    204     self._data_attr.extend(['pinv_wexog', 'wendog', 'wexog', 'weights'])

File /opt/conda/lib/python3.10/site-packages/statsmodels/base/model.py:270, in␣
↪LikelihoodModel.__init__(self, endog, exog, **kwargs)
    269 def __init__(self, endog, exog=None, **kwargs):
--> 270     super().__init__(endog, exog, **kwargs)
    271     self.initialize()
```

```
File /opt/conda/lib/python3.10/site-packages/statsmodels/base/model.py:95, in␣
 ↪Model.__init__(self, endog, exog, **kwargs)
     93 missing = kwargs.pop('missing', 'none')
     94 hasconst = kwargs.pop('hasconst', None)
---> 95 self.data = self._handle_data(endog, exog, missing, hasconst,
     96                               **kwargs)
     97 self.k_constant = self.data.k_constant
     98 self.exog = self.data.exog

File /opt/conda/lib/python3.10/site-packages/statsmodels/base/model.py:135, in␣
 ↪Model._handle_data(self, endog, exog, missing, hasconst, **kwargs)
    134 def _handle_data(self, endog, exog, missing, hasconst, **kwargs):
--> 135     data = handle_data(endog, exog, missing, hasconst, **kwargs)
    136     # kwargs arrays could have changed, easier to just attach here
    137     for key in kwargs:

File /opt/conda/lib/python3.10/site-packages/statsmodels/base/data.py:675, in␣
 ↪handle_data(endog, exog, missing, hasconst, **kwargs)
    672     exog = np.asarray(exog)
    674 klass = handle_data_class_factory(endog, exog)
--> 675 return klass(endog, exog=exog, missing=missing, hasconst=hasconst,
    676              **kwargs)

File /opt/conda/lib/python3.10/site-packages/statsmodels/base/data.py:84, in␣
 ↪ModelData.__init__(self, endog, exog, missing, hasconst, **kwargs)
     82     self.orig_endog = endog
     83     self.orig_exog = exog
---> 84     self.endog, self.exog = self._convert_endog_exog(endog, exog)
     86 self.const_idx = None
     87 self.k_constant = 0

File /opt/conda/lib/python3.10/site-packages/statsmodels/base/data.py:509, in␣
 ↪PandasData._convert_endog_exog(self, endog, exog)
    507 exog = exog if exog is None else np.asarray(exog)
    508 if endog.dtype == object or exog is not None and exog.dtype == object:
--> 509     raise ValueError("Pandas data cast to numpy dtype of object. "
    510                      "Check input data with np.asarray(data).")
    511 return super(PandasData, self)._convert_endog_exog(endog, exog)

ValueError: Pandas data cast to numpy dtype of object. Check input data with np
 ↪asarray(data).
```

[42]:

```
#hyperperameter tuning
grid_vals = {'penalty': ['l1','l2'], 'C': [0.001,0.005,0.01,0.05,0.1,0.5,1,5]}
#lasso
lasso_param_grid = {
```

```
    'alpha': [0.001, 0.01, 0.1, 1.0, 10.0, 100.0]
}
lasso_model = Lasso()
lasso_grid_search = GridSearchCV(lasso_model, lasso_param_grid, cv=5,␣
 ↪scoring='neg_mean_squared_error')
lasso_grid_search.fit(X_train, y_train)
best_lasso_alpha = lasso_grid_search.best_params_['alpha']
best_lasso_model = lasso_grid_search.best_estimator_

# Evaluate Lasso model
lasso_predictions = best_lasso_model.predict(X_test)
r2_test = r2_score(y_test, lasso_predictions)
res=stats.spearmanr(y_test, lasso_predictions)
print('R squared value : ', r2_test)
print(f"Lasso Best Alpha: {best_lasso_alpha}")
print(f"Lasso Spearman Rank: {res.statistic}")
#ridge

ridge_param_grid = {
    'alpha': [0.001, 0.01, 0.1, 1.0, 10.0, 100.0]
}

# Create Ridge regression model
ridge_model = Ridge()

# Perform grid search for Ridge regression
ridge_grid_search = GridSearchCV(ridge_model, ridge_param_grid, cv=5,␣
 ↪scoring='neg_mean_squared_error')
ridge_grid_search.fit(X_train, y_train)

# Get best hyperparameters and corresponding model
best_ridge_alpha = ridge_grid_search.best_params_['alpha']
best_ridge_model = ridge_grid_search.best_estimator_

# Evaluate Ridge model
ridge_predictions = best_ridge_model.predict(X_test)
ridge_r2 = r2_score(y_test, ridge_predictions)
res=stats.spearmanr(y_test, ridge_predictions)
print(f"Ridge Best Alpha: {best_ridge_alpha}")
print(f"Ridge R2: {ridge_r2}")
print(f"Ridge Spearman Rank: {res.statistic}")
```

```
R squared value :  0.016269223581383607
Lasso Best Alpha: 0.01
Lasso Spearman Rank: 0.14957908212881274
Ridge Best Alpha: 100.0
Ridge R2: 0.017031388436248185
```

```
Ridge Spearman Rank: 0.15279010365249476
```

4 Decision Tree - Regression: whether data collected about disease can increase the cost of insurance products

[65]:
```python
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeRegressor
from sklearn.metrics import mean_squared_error, r2_score
```

Correlation Analysis: Calculate the correlation between each feature and the target variable (insurance costs). Features with higher absolute correlation values are more likely to be strong predictors.

[53]:
```python
#drop id from merged_df for correaltion analysis - string
#merged_df.drop('Premium')
corr_analysis = merged_df.drop(['ID', 'Age','Health_Condition_ID_x', 'Weight',
                               'Height', 'Pregnant', 'Health_Condition_ID_y',
                               'disease', 'Life_Style_ID', 'DRKSTAT_A',␣
 ↪'ID_1', 'Life_Style_ID_1',
                               'SMKCIGST_A', 'ID_2',␣
 ↪'Life_Style_ID_2','Premium','LogPremium', 'BMI'], axis=1)
```

[54]:
```python
corr_analysis
```

[54]:
```
       Urban_Rural  Region  Gender  Race  Education  Num_Fam_Adult  \
3                1       1       1     1          8              1
4                1       4       2     1          5              3
5                1       4       2     1          5              2
13               1       2       1     1          5              2
15               1       1       1     2          8              2
…              …     …     …   …        …            …
27643            3       2       1     6          3              2
27644            1       3       2     6          1              2
27645            1       3       2     6          8              3
27648            3       2       2     6          5              3
27650            2       4       2     6          9              2

       Num_Fam_Kid  Current_MaritalStatus  Citizenship  JobYN  …  Walking  \
3                0                      9            7   <NA>  …        1
4                1                      1            1   <NA>  …        1
5                1                      7            1   <NA>  …        1
13               0                      9            7   <NA>  …        1
15               3                      9            7   <NA>  …        1
…              …                    …          …    …  …      …
27643            0                      9            8   <NA>  …        1
27644            3                      1            8   <NA>  …        1
27645            2                      9            8   <NA>  …        1
27648            0                      9            8   <NA>  …        2
```

```
27650            0                      9            8    <NA>  …         1

       Sleeping  Eating  Meditation  Yoga  Therapy  Dr_Visit  Coverage  \
3             6       1           2     7        2         0         2
4             8       1           1     1        1         0         2
5             8       2           2     2        2         0         2
13            8       1           2     2        2         0         2
15            6       1           2     2        2         0         2
…           …      …          …    …      …       …        …
27643        98       8           8     8        8         8         2
27644         7       1           8     8        2         0         2
27645        98       8           8     8        8         8         2
27648        98       8           8     8        1         5         2
27650        98       7           8     8        2         0         2

       ProductType  smoke
3                1      0
4                1      0
5                1      0
13               1      0
15               1      0
…              …    …
27643            1      2
27644            1      0
27645            1      2
27648            1      2
27650            1      2

[12928 rows x 39 columns]
```

[55]:
```python
#categorical variables in list
cat_vars=['Urban_Rural','Region','Gender','Race','Education','Num_Fam_Adult','Num_Fam_Kid',
          'Current_MaritalStatus','Citizenship','JobYN','Housing','Health_WeakImmune',
          'Hypertension','High_Cholesterol','CoronaryHeartDisease','Angina','HeartAttack',
          'Stroke','Asthma','Cancer','Prediabetes','GestationalDiabetes','COPD',
          'Arthritis','Dementia','Anxiety_Disorder','Depression','Epilepsy',
          'Chronic_Fatigue_Syndrome','Walking','Sleeping','Eating','Meditation',
          'Yoga','Therapy','Dr_Visit','Coverage','ProductType','smoke']

# Perform one-hot encoding on the categorical variables
data_encoded = pd.get_dummies(corr_analysis, columns=cat_vars)
selected_columns = ['Age', 'BMI', 'LogPremium']
combined_corr_analysis = pd.concat([data_encoded, merged_df[selected_columns]],
    axis=1)
combined_corr_analysis
```

```
[55]:        Urban_Rural_1  Urban_Rural_2  Urban_Rural_3  Urban_Rural_4  Region_1  \
3                True          False          False          False      True
4                True          False          False          False     False
5                True          False          False          False     False
13               True          False          False          False     False
15               True          False          False          False      True
…                 …             …             …             …         …
27643           False          False           True          False     False
27644            True          False          False          False     False
27645            True          False          False          False     False
27648           False          False           True          False     False
27650           False           True          False          False     False

       Region_2  Region_3  Region_4  Gender_1  Gender_2  …  Dr_Visit_8  \
3         False     False     False      True     False  …       False
4         False     False      True     False      True  …       False
5         False     False      True     False      True  …       False
13         True     False     False      True     False  …       False
15        False     False     False      True     False  …       False
…           …         …         …         …         …  …         …
27643      True     False     False      True     False  …        True
27644     False      True     False     False      True  …       False
27645     False      True     False     False      True  …        True
27648      True     False     False     False      True  …       False
27650     False     False      True     False      True  …       False

       Dr_Visit_9  Coverage_2  ProductType_1  smoke_0  smoke_1  smoke_2  Age  \
3           False        True           True     True    False    False   54
4           False        True           True     True    False    False   44
5           False        True           True     True    False    False   19
13          False        True           True     True    False    False   31
15          False        True           True     True    False    False   34
…             …          …              …        …        …       …   …
27643       False        True           True    False    False     True   42
27644       False        True           True     True    False    False   27
27645       False        True           True    False    False     True   41
27648       False        True           True    False    False     True   20
27650       False        True           True    False    False     True   58

              BMI  LogPremium
3       25.678217   11.512915
4       36.355221    8.881836
5       74.498923   11.512915
13      28.974799   11.512915
15     147.524997    8.476371
…             …          …
27643   28.480828   11.512915
```

```
27644   27.341595    6.49224
27645   29.228138    8.38754
27648   31.093088   11.512915
27650  156.150673    8.476371

[12928 rows x 198 columns]
```

[56]:
```python
# Calculate correlation matrix for the combined DataFrame
correlation_matrix = combined_corr_analysis.corr()
correlation_matrix
```

[56]:
```
               Urban_Rural_1  Urban_Rural_2  Urban_Rural_3  Urban_Rural_4  \
Urban_Rural_1       1.000000      -0.420660      -0.444056      -0.267079
Urban_Rural_2      -0.420660       1.000000      -0.370404      -0.222781
Urban_Rural_3      -0.444056      -0.370404       1.000000      -0.235171
Urban_Rural_4      -0.267079      -0.222781      -0.235171       1.000000
Region_1           -0.044503       0.135337      -0.014155      -0.097038
…                        …              …              …              …
smoke_1            -0.057710      -0.019006       0.025375       0.073360
smoke_2             0.012320       0.001114      -0.007259      -0.009230
Age                -0.086709       0.040062       0.018043       0.046301
BMI                -0.025288      -0.013311       0.020114       0.026504
LogPremium         -0.006993       0.037822      -0.014796      -0.020129

               Region_1  Region_2  Region_3  Region_4  Gender_1  Gender_2  \
Urban_Rural_1 -0.044503 -0.125646 -0.042243  0.206387  0.015638 -0.015636
Urban_Rural_2  0.135337  0.022134  0.019841 -0.161465 -0.008538  0.008813
Urban_Rural_3 -0.014155  0.002181 -0.003592  0.014225 -0.015806  0.015407
Urban_Rural_4 -0.097038  0.147587  0.039019 -0.100226  0.010523 -0.010348
Region_1       1.000000 -0.247280 -0.336761 -0.265663 -0.005315  0.005119
…                   …         …         …         …         …         …
smoke_1       -0.002024  0.039853  0.016917 -0.055223  0.045345 -0.045198
smoke_2        0.021704 -0.017579  0.008606 -0.011513 -0.000896  0.000966
Age            0.019258 -0.002263  0.017176 -0.033552 -0.036428  0.036630
BMI            0.018125  0.005142  0.007282 -0.028810 -0.059517  0.058583
LogPremium     0.030337  0.027180 -0.020084 -0.030607  0.007095 -0.007103

               …  Dr_Visit_8  Dr_Visit_9  Coverage_2  ProductType_1  \
Urban_Rural_1  …    0.010479   -0.014577         NaN            NaN
Urban_Rural_2  …    0.003528   -0.004644         NaN            NaN
Urban_Rural_3  …   -0.011515    0.019124         NaN            NaN
Urban_Rural_4  …   -0.003999    0.000974         NaN            NaN
Region_1       …   -0.010874   -0.009647         NaN            NaN
…             …          …           …           …              …
smoke_1        …   -0.024051    0.005133         NaN            NaN
smoke_2        …    0.505168    0.025586         NaN            NaN
Age            …    0.004033   -0.018286         NaN            NaN
```

```
BMI             …    0.009127   -0.005180         NaN            NaN
LogPremium      …    0.019294    0.026675         NaN            NaN

                 smoke_0   smoke_1   smoke_2       Age       BMI  LogPremium
Urban_Rural_1   0.046546 -0.057710  0.012320 -0.086709 -0.025288   -0.006993
Urban_Rural_2   0.016706 -0.019006  0.001114  0.040062 -0.013311    0.037822
Urban_Rural_3  -0.019604  0.025375 -0.007259  0.018043  0.020114   -0.014796
Urban_Rural_4  -0.062176  0.073360 -0.009230  0.046301  0.026504   -0.020129
Region_1       -0.008318 -0.002024  0.021704  0.019258  0.018125    0.030337
…                    …         …         …         …         …           …
smoke_1        -0.884129  1.000000 -0.047610  0.065610 -0.003962   -0.028481
smoke_2        -0.424619 -0.047610  1.000000  0.002776  0.031881    0.045684
Age            -0.060767  0.065610  0.002776  1.000000  0.046816   -0.016496
BMI            -0.011322 -0.003962  0.031881  0.046816  1.000000    0.018190
LogPremium      0.004446 -0.028481  0.045684 -0.016496  0.018190    1.000000

[198 rows x 198 columns]
```

[60]:

```python
correlation_matrix['LogPremium'].abs()
```

[60]:

```
Urban_Rural_1    0.006993
Urban_Rural_2    0.037822
Urban_Rural_3    0.014796
Urban_Rural_4    0.020129
Region_1         0.030337
                   …
smoke_1          0.028481
smoke_2          0.045684
Age              0.016496
BMI              0.018190
LogPremium       1.000000
Name: LogPremium, Length: 198, dtype: float64
```

[75]:

```python
# Assuming 'data' is your DataFrame with variables and target variable
correlation_matrix = combined_corr_analysis.corr()
x=correlation_matrix['LogPremium'].abs()
correlation_with_target = correlation_matrix['LogPremium'].abs().
  ↪sort_values(ascending=False)

# Select the top N features with the highest correlation coefficients (e.g.,␣
  ↪top 10)
top_n_features = correlation_with_target.head(11).index.tolist()

print("Top N features with the highest correlation to 'LogPremium':")
print(top_n_features)
```

```
Top N features with the highest correlation to 'LogPremium':
['LogPremium', 'Num_Fam_Adult_1', 'JobYN_1', 'Current_MaritalStatus_1',
```

```
'Current_MaritalStatus_5', 'Housing_2', 'Num_Fam_Kid_0', 'Num_Fam_Adult_3',
'Current_MaritalStatus_9', 'Housing_1', 'Citizenship_2']
```

[78]:
```python
# Split the data into features (X) and target variable (y)
x = combined_corr_analysis[['Num_Fam_Adult_1', 'JobYN_1',␣
 ↪'Current_MaritalStatus_1', 'Current_MaritalStatus_5', 'Housing_2',␣
 ↪'Num_Fam_Kid_0', 'Num_Fam_Adult_3', 'Current_MaritalStatus_9', 'Housing_1',␣
 ↪'Citizenship_2']]
y = combined_corr_analysis[['LogPremium']]

# Split the data into training and testing sets (80% training, 20% testing)
X_train, X_test, y_train, y_test = train_test_split(x, y, test_size=0.2,␣
 ↪random_state=42)

# Create a Decision Tree Regressor model
dt_model = DecisionTreeRegressor()

# Train the model on the training data
dt_model.fit(X_train, y_train)

# Make predictions on the test data
y_pred = dt_model.predict(X_test)
```

[79]:
```python
# Calculate mean squared error (MSE) and R-squared (R2) for evaluation
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
res=  stats.spearmanr(y_test, y_pred)
print("Mean Squared Error:", mse)
print("R-squared:", r2)
print('Spearman Rank:', res.statistic)
```

```
Mean Squared Error: 2.5676576333913843
R-squared: 0.019967675914570l6
Spearman Rank: 0.1929283094629408
```

[83]:
```python
#serialize model:
trained_model = pickle.dumps(dt_model,'test.pickle')
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[83], line 2
      1 #serialize model:
----> 2 trained_model = pickle.dumps(dt_model,'test.pickle')

TypeError: 'str' object cannot be interpreted as an integer
```

[ ]:
```python
pickle.dump(dt_model, open(filename, 'wb'))
```

```
[ ]:
```