In [3]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
```

In [4]:
```
df=pd.read_csv(r"C:\Users\DELL\Downloads\car_insurance_premium_dataset.csv")
```

In [5]:
```
df.head()
```

Out[5]:

	Driver Age	Driver Experience	Previous Accidents	Annual Mileage (x1000 km)	Car Manufacturing Year	Car Age	Insurance Premium ($)
0	56	32	4	17	2002	23	488.35
1	46	19	0	21	2025	0	486.15
2	32	11	4	15	2020	5	497.55
3	60	0	4	19	1991	34	498.35
4	25	7	0	13	2005	20	495.55

In [6]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 7 columns):
 #   Column                     Non-Null Count  Dtype  
---  ------                     --------------  -----  
 0   Driver Age                 1000 non-null   int64  
 1   Driver Experience          1000 non-null   int64  
 2   Previous Accidents         1000 non-null   int64  
 3   Annual Mileage (x1000 km)  1000 non-null   int64  
 4   Car Manufacturing Year     1000 non-null   int64  
 5   Car Age                    1000 non-null   int64  
 6   Insurance Premium ($)      1000 non-null   float64
dtypes: float64(1), int64(6)
memory usage: 54.8 KB
```

In [7]:
```
df.describe()
```

Out[7]:

	Driver Age	Driver Experience	Previous Accidents	Annual Mileage (x1000 km)	Car Manufacturing Year	Car Age
count	1000.000000	1000.000000	1000.0000	1000.000000	1000.000000	1000.000000
mean	41.575000	14.759000	2.5680	17.933000	2007.637000	17.363000
std	13.765677	10.544292	1.6989	4.410665	10.363331	10.363331
min	18.000000	0.000000	0.0000	11.000000	1990.000000	0.000000
25%	30.000000	6.000000	1.0000	14.000000	1999.000000	8.000000
50%	42.000000	13.000000	3.0000	18.000000	2008.000000	17.000000
75%	53.000000	23.000000	4.0000	22.000000	2017.000000	26.000000
max	65.000000	40.000000	5.0000	25.000000	2025.000000	35.000000

In [8]:
```
df.isnull().sum()
```

Out[8]:
```
Driver Age                   0
Driver Experience            0
Previous Accidents           0
Annual Mileage (x1000 km)    0
Car Manufacturing Year       0
Car Age                      0
Insurance Premium ($)        0
dtype: int64
```

In [9]:
```
df.shape
```

Out[9]:
```
(1000, 7)
```

In [10]:
```
df.duplicated()
```

Out[10]:
```
0      False
1      False
2      False
3      False
4      False
       ...
995    False
996    False
997    False
998    False
999    False
Length: 1000, dtype: bool
```

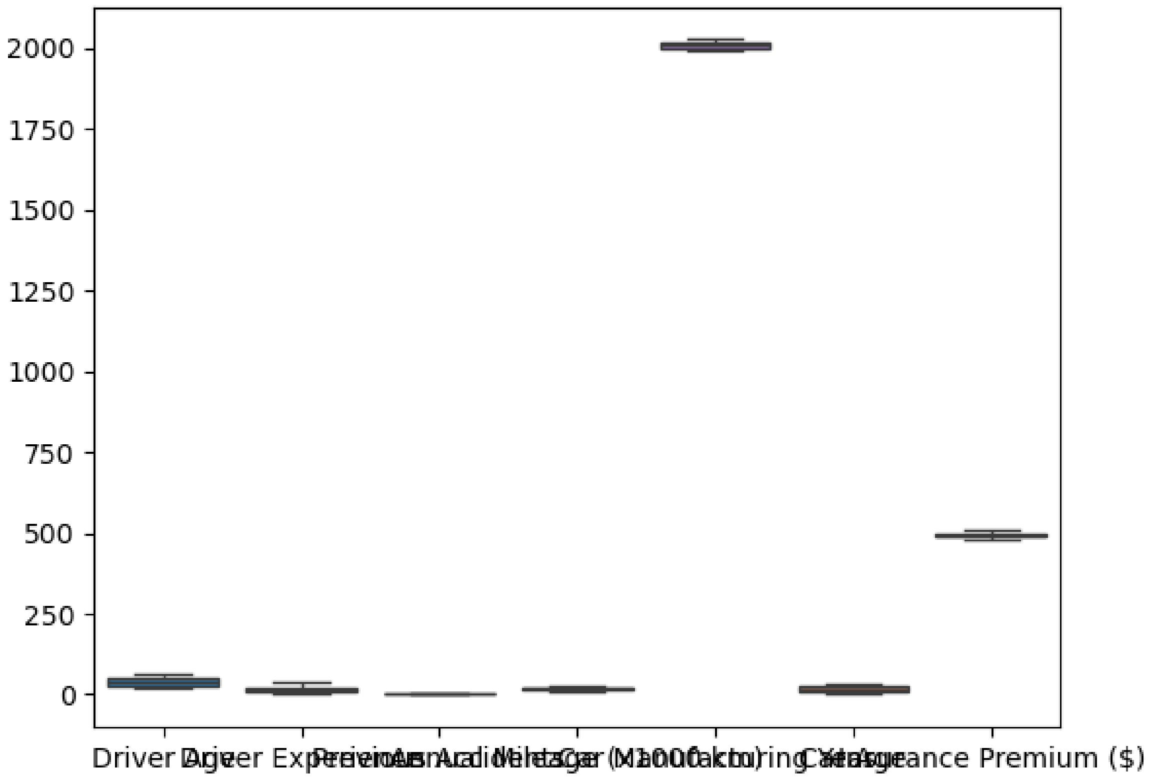
In [11]:
```
df.head()
```

Out[11]:

	Driver Age	Driver Experience	Previous Accidents	Annual Mileage (x1000 km)	Car Manufacturing Year	Car Age	Insurance Premium ($)
0	56	32	4	17	2002	23	488.35
1	46	19	0	21	2025	0	486.15
2	32	11	4	15	2020	5	497.55
3	60	0	4	19	1991	34	498.35
4	25	7	0	13	2005	20	495.55

In [12]:
```
X = df.drop(['Insurance Premium ($)'], axis=1)
y = df['Insurance Premium ($)']
```

In [13]:
```
sns.boxplot(df)
plt.show()
plt.figure(figsize=(10,30))
```



Out[13]:
```
<Figure size 1000x3000 with 0 Axes>
```
```
<Figure size 1000x3000 with 0 Axes>
```

In [14]:
```
def outliertreat(df,col):
    Q1 = df[col].quantile(0.25)
    Q3 = df[col].quantile(0.75)
    IQR = Q3 - Q1
    UL = Q3 + IQR
    LL = Q1 - IQR
    df.loc[df[col]>UL,col] = df[col].median()
    df.loc[df[col]<LL,col] = df[col].median()
```

```
In [15]: outliertreat(df,"Driver Age")
```

```
In [16]: outliertreat(df,"Driver Experience")
```

```
In [17]: outliertreat(df,"Previous Accidents")
```

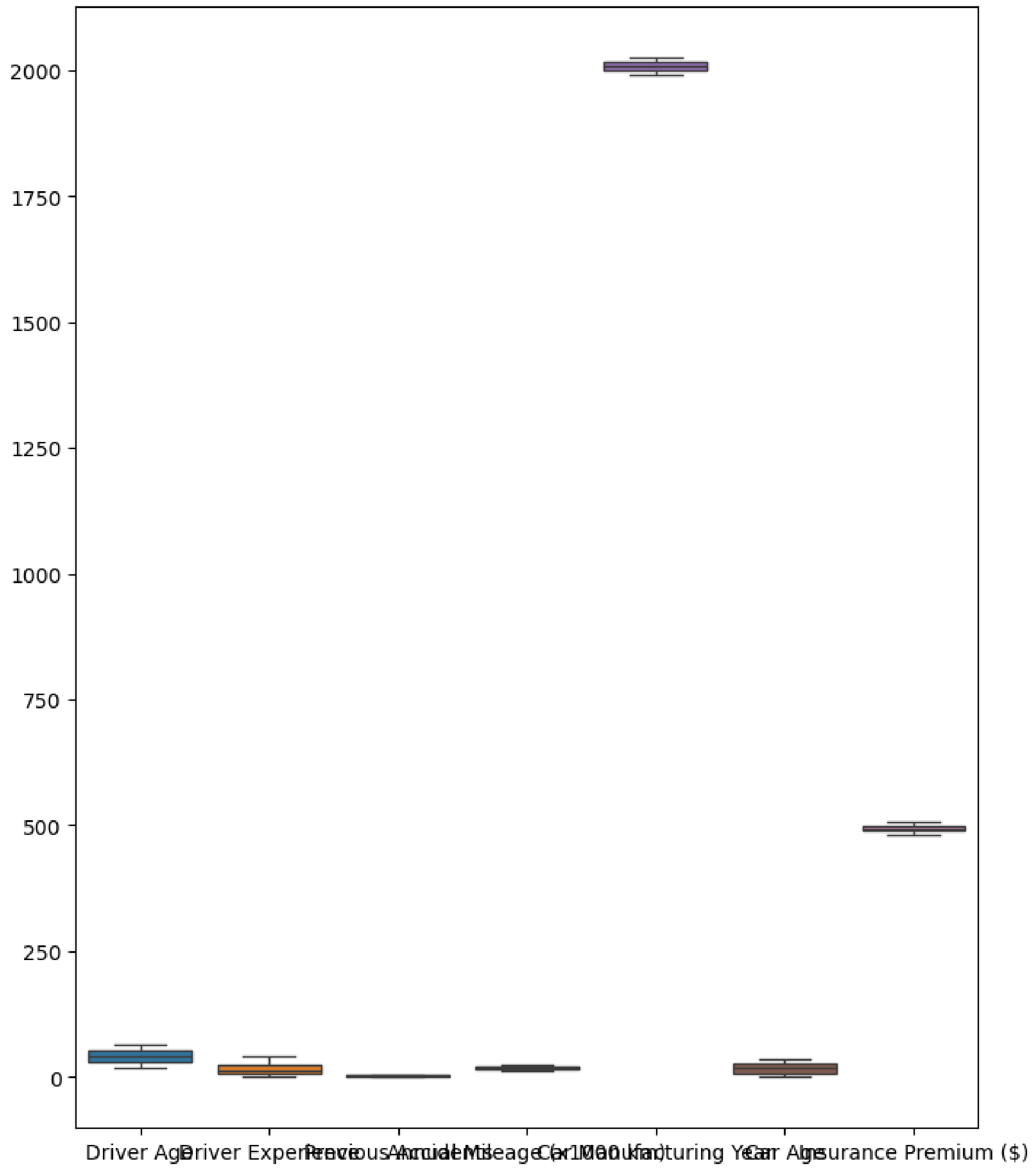
```
In [18]: outliertreat(df,"Annual Mileage (x1000 km)")
```

```
In [19]: outliertreat(df,"Car Manufacturing Year")
```

```
In [20]: outliertreat(df,"Car Age")
```

```
In [21]: outliertreat(df,"Insurance Premium ($)")
```

```
In [22]: plt.figure(figsize=(8,10))
         sns.boxplot(df)
         plt.show()
```



```
In [23]: from sklearn.preprocessing import StandardScaler
```

```
In [24]: SS=StandardScaler()
```

```
In [25]: from sklearn.model_selection import train_test_split
```

```
In [26]: df["Car Manufacturing Year"]=SS.fit_transform(df[["Car Manufacturing Year"]])
```

```
In [27]: df["Driver Age"]=SS.fit_transform(df[["Driver Age"]])
```

```
In [28]: df["Driver Experience"]=SS.fit_transform(df[["Driver Experience"]])
```

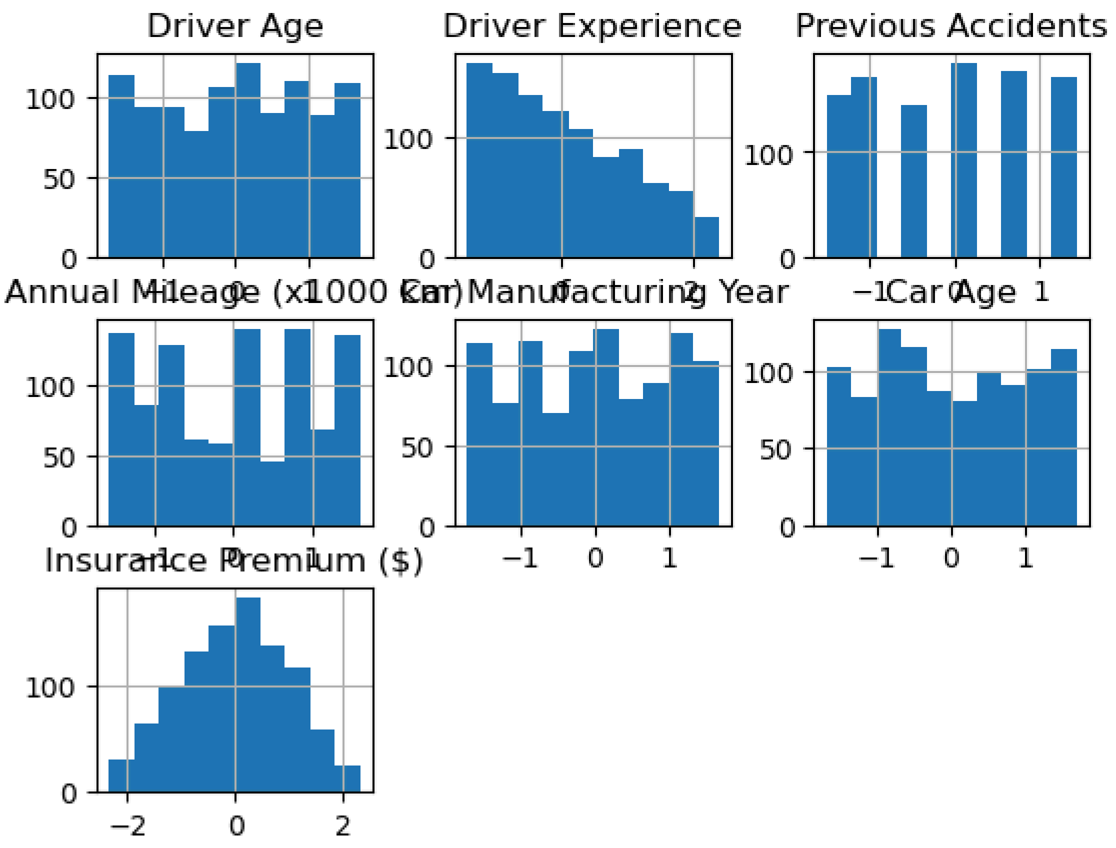
```
In [29]: df["Previous Accidents"]=SS.fit_transform(df[["Previous Accidents"]])
```

```
In [30]: df["Annual Mileage (x1000 km)"]=SS.fit_transform(df[["Annual Mileage (x1000 km)"
```

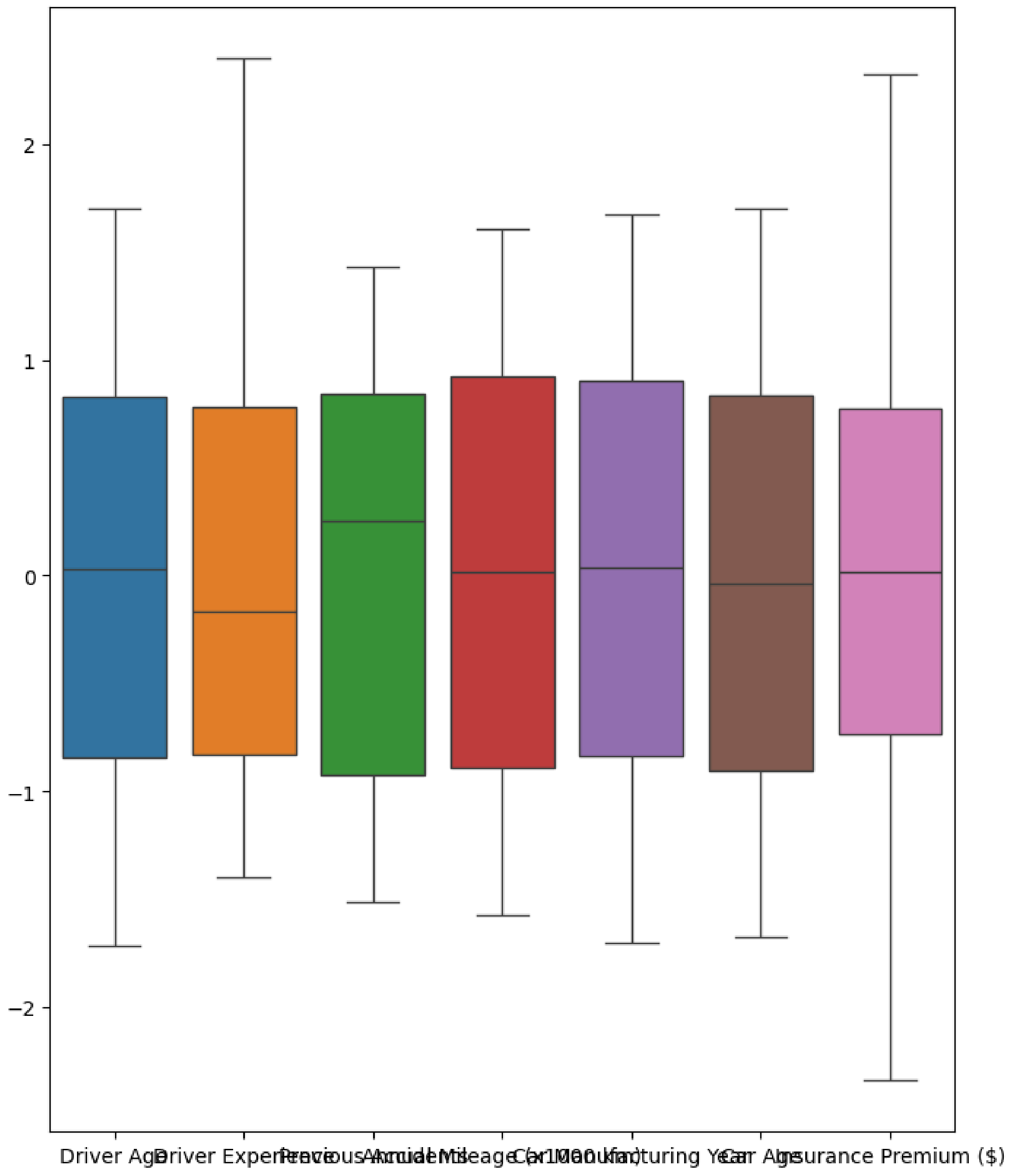
```
In [31]: df["Car Age"]=SS.fit_transform(df[["Car Age"]])
```

```
In [32]: df["Insurance Premium ($)"]=SS.fit_transform(df[["Insurance Premium ($)"]])
```

```
In [33]: df.hist()
         plt.show()
```



```
In [34]: plt.figure(figsize=(8,10))
         sns.boxplot(df)
         plt.show()
```



```
In [35]: df.skew(numeric_only=True)
```

```
Out[35]: Driver Age                  -0.047599
         Driver Experience            0.446676
         Previous Accidents          -0.064745
         Annual Mileage (x1000 km)    0.019283
         Car Manufacturing Year      -0.037801
         Car Age                      0.037801
         Insurance Premium ($)       -0.081066
         dtype: float64
```

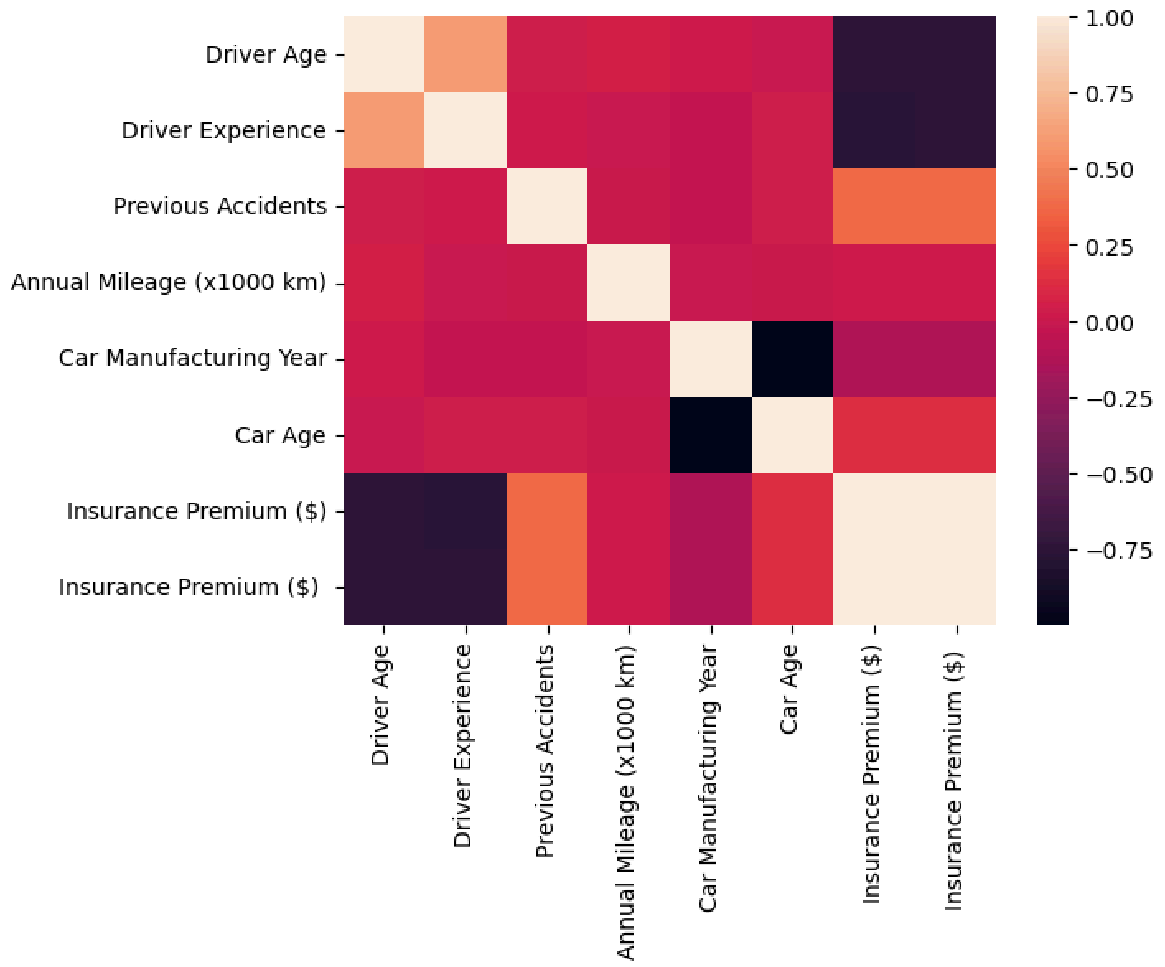
```
In [36]: from sklearn.preprocessing import PowerTransformer
```

```
In [37]: PT=PowerTransformer()
```

```
In [38]: df[['Driver Age','Previous Accidents','Annual Mileage (x1000 km)','Insurance Pre
```

```
In [40]: sns.heatmap(df.corr(numeric_only=True))
```

```
Out[40]: <Axes: >
```



```
In [44]: from sklearn.preprocessing import LabelEncoder
```

```
In [49]: LE=LabelEncoder()
```

```
In [52]: df['Driver Experience']=LE.fit_transform(df['Driver Experience'])
```

```
In [53]: df['Previous Accidents']=LE.fit_transform(df['Previous Accidents'])
```

```
In [54]: df.head()
```

Out[54]:

	Driver Age	Driver Experience	Previous Accidents	Annual Mileage (x1000 km)	Car Manufacturing Year	Car Age	Insurance Premium ($)	In P
0	1.051806	32	4	-0.204619	-0.544209	0.553817	-0.985432	-(
1	0.305622	19	0	0.699163	1.676265	-1.700524	-1.379856	-1
2	-0.704576	11	4	-0.661633	1.193553	-1.200460	0.663977	(
3	1.354392	0	4	0.248952	-1.606175	1.586226	0.807403	(
4	-1.196660	7	0	-1.121210	-0.254582	0.267939	0.305410	(

```
In [55]: from sklearn.model_selection import train_test_split
```

```
In [56]: X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.25,random_state=4
```

```
In [66]: from sklearn.linear_model import LinearRegression
         from sklearn.tree import DecisionTreeRegressor
         from sklearn.neighbors import KNeighborsRegressor
         from sklearn.ensemble import RandomForestRegressor, GradientBoostingRegressor
```

```
In [67]: LR=LinearRegression()
```

```
In [68]: LR.fit(X_train,y_train)
```

```
Out[68]: ▾ LinearRegression
         LinearRegression()
```

```
In [70]: LR_pred = LR.predict(X_test)
```

```
In [71]: from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score
```

```
In [74]: mean_absolute_error(y_test,LR_pred)
```

```
Out[74]: 0.6982000000000009
```

```
In [75]: mean_squared_error(y_test,LR_pred)
```

```
Out[75]: 2.952095546874999
```

```
In [80]: r2_score(y_test, LR_pred)
```

```
Out[80]: 0.8980582686377323
```

```
In [85]: LR.score(X_train,y_train)*100
```

```
Out[85]: 89.06312739785875
```

```
In [86]: LR.score(X_test,y_test)
```

```
Out[86]: 0.8980582686377323
```

```
In [87]: KNR = KNeighborsRegressor(n_neighbors=7)
```

```
In [88]: KNR.fit(X_train,y_train)
```

```
Out[88]: ▾ KNeighborsRegressor
         KNeighborsRegressor(n_neighbors=7)
```

```
In [89]: KNR_pred = KNR.predict(X_test)
```

```
In [90]: mean_absolute_error(y_test,KNR_pred)
```

Out[90]:
```
1.9589428571428553
```

In [92]:
```
mean_squared_error(y_test,LR_pred)
```

Out[92]:
```
2.952095546874999
```

In [93]:
```
r2_score(y_test, KNR_pred)
```

Out[93]:
```
0.7866936267954454
```

In [99]:
```
KNR.score(X_train,y_train)*100
```

Out[99]:
```
84.46056880549351
```

In [100…
```
KNR.score(X_test,y_test)
```

Out[100…
```
0.7866936267954454
```

In [101…
```
RFR=RandomForestRegressor(n_estimators=100, random_state=42)
```

In [102…
```
RFR.fit(X_train, y_train)
```

Out[102…
```
▾ RandomForestRegressor
RandomForestRegressor(random_state=42)
```

In [105…
```
RFR_pred = RFR.predict(X_test)
```

In [112…
```
mean_absolute_error(y_test, RFR_pred)
```

Out[112…
```
0.8007759999999993
```

In [113…
```
mean_squared_error(y_test, RFR_pred)
```

Out[113…
```
2.3414879419999868
```

In [114…
```
r2_score(y_test, RFR_pred)
```

Out[114…
```
0.919143763817527
```

In [115…
```
RFR.score(X_train,y_train)*100
```

Out[115…
```
99.04901477218021
```

In [116…
```
RFR.score(X_test,y_test)
```

Out[116…
```
0.919143763817527
```

In [117…
```
DTC = DecisionTreeRegressor(random_state=42)
```

In [120…
```
DTC.fit(X_train, y_train)
```

```
Out[120…
▾ DecisionTreeRegressor
DecisionTreeRegressor(random_state=42)
```

```
In [124… y_pred_dt = DTC.predict(X_test)
```

```
In [127… mean_absolute_error(y_test, y_pred_dt)
Out[127… 1.3024000000000013
```

```
In [132… mean_squared_error(y_test, y_pred_dt)
Out[132… 3.692720000000007
```

```
In [138… r2_score(y_test,y_pred_dt )
Out[138… 0.872483033066269
```

```
In [142… DTC.score(X_train,y_train)*100
Out[142… 100.0
```

```
In [143… DTC.score(X_test,y_test)
Out[143… 0.872483033066269
```

```
In [145… GBL = GradientBoostingRegressor(random_state=42, n_estimators=100, learning_rate
```

```
In [147… GBL.fit(X_train, y_train)
Out[147…
▾ GradientBoostingRegressor
GradientBoostingRegressor(random_state=42)
```

```
In [155… gbl_pred = GBL.predict(X_test)
```

```
In [158… mean_squared_error(y_test, gbl_pred)
Out[158… 2.280836315268464
```

```
In [161… mean_absolute_error(y_test, gbl_pred)
Out[161… 0.7619309738786116
```

```
In [165… r2_score(y_test,gbl_pred)
Out[165… 0.9212381851330886
```

```
In [167… GBL.score(X_train,y_train)*100
Out[167… 98.21631826760195
```

```
In [170… GBL.score(X_test,y_test)
```

```
Out[170…   0.9212381851330886
```

```
In [175…   from sklearn.model_selection import GridSearchCV, RandomizedSearchCV
```

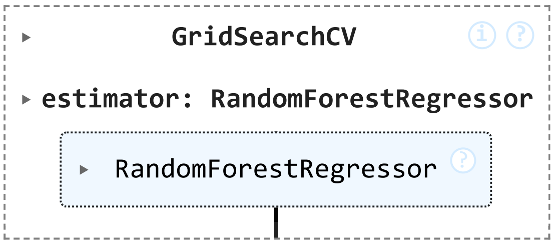
```
In [177…   param_grid = {
               'n_estimators': [50, 100, 200],
               'max_depth': [None, 10, 20, 30],
           }
```

```
In [179…   rf_model = RandomForestRegressor(random_state=42)
```

```
In [181…   grid_search = GridSearchCV(
               estimator=RandomForestRegressor(random_state=42),
               param_grid=param_grid,
           )
```

```
In [183…   grid_search.fit(X_train, y_train)
```

Out[183…



```
In [184…   grid_search.best_params_
```

```
Out[184…   {'max_depth': 20, 'n_estimators': 200}
```

```
In [185…   Best_GS=grid_search.best_estimator_
```

```
In [186…   y_pred=Best_GS.predict(X_test)
```

```
In [187…   print("Best MSE:", -grid_search.best_score_)
```

```
Best MSE: -0.9297245088699227
```

```
In [188…   best_rf_model = grid_search.best_estimator_
           rf_pred = best_rf_model.predict(X_test)
```

```
In [189…   rf_r2 = r2_score(y_test, rf_pred)
```

```
In [190…   print("Tuned Random Forest - MSE:",  "R²:", rf_r2)
```

```
Tuned Random Forest - MSE: R²: 0.9230631762565836
```

```
In [191…   pip install --upgrade gradio
```

```
Requirement already satisfied: gradio in c:\users\dell\anaconda3\lib\site-package
s (5.13.1)
Requirement already satisfied: aiofiles<24.0,>=22.0 in c:\users\dell\anaconda3\li
b\site-packages (from gradio) (23.2.1)
Requirement already satisfied: anyio<5.0,>=3.0 in c:\users\dell\anaconda3\lib\sit
e-packages (from gradio) (4.2.0)
Requirement already satisfied: fastapi<1.0,>=0.115.2 in c:\users\dell\anaconda3\l
ib\site-packages (from gradio) (0.115.6)
Requirement already satisfied: ffmpy in c:\users\dell\anaconda3\lib\site-packages
(from gradio) (0.5.0)
Requirement already satisfied: gradio-client==1.6.0 in c:\users\dell\anaconda3\li
b\site-packages (from gradio) (1.6.0)
Requirement already satisfied: httpx>=0.24.1 in c:\users\dell\anaconda3\lib\site-
packages (from gradio) (0.26.0)
Requirement already satisfied: huggingface-hub>=0.25.1 in c:\users\dell\anaconda3
\lib\site-packages (from gradio) (0.27.1)
Requirement already satisfied: jinja2<4.0 in c:\users\dell\anaconda3\lib\site-pac
kages (from gradio) (3.1.4)
Requirement already satisfied: markupsafe~=2.0 in c:\users\dell\anaconda3\lib\sit
e-packages (from gradio) (2.1.3)
Requirement already satisfied: numpy<3.0,>=1.0 in c:\users\dell\anaconda3\lib\sit
e-packages (from gradio) (1.26.4)
Requirement already satisfied: orjson~=3.0 in c:\users\dell\anaconda3\lib\site-pa
ckages (from gradio) (3.10.15)
Requirement already satisfied: packaging in c:\users\dell\anaconda3\lib\site-pack
ages (from gradio) (23.2)
Requirement already satisfied: pandas<3.0,>=1.0 in c:\users\dell\anaconda3\lib\si
te-packages (from gradio) (2.2.2)
Requirement already satisfied: pillow<12.0,>=8.0 in c:\users\dell\anaconda3\lib\s
ite-packages (from gradio) (10.3.0)
Requirement already satisfied: pydantic>=2.0 in c:\users\dell\anaconda3\lib\site-
packages (from gradio) (2.5.3)
Requirement already satisfied: pydub in c:\users\dell\anaconda3\lib\site-packages
(from gradio) (0.25.1)
Requirement already satisfied: python-multipart>=0.0.18 in c:\users\dell\anaconda
3\lib\site-packages (from gradio) (0.0.20)
Requirement already satisfied: pyyaml<7.0,>=5.0 in c:\users\dell\anaconda3\lib\si
te-packages (from gradio) (6.0.1)
Requirement already satisfied: ruff>=0.2.2 in c:\users\dell\anaconda3\lib\site-pa
ckages (from gradio) (0.9.2)
Requirement already satisfied: safehttpx<0.2.0,>=0.1.6 in c:\users\dell\anaconda3
\lib\site-packages (from gradio) (0.1.6)
Requirement already satisfied: semantic-version~=2.0 in c:\users\dell\anaconda3\l
ib\site-packages (from gradio) (2.10.0)
Requirement already satisfied: starlette<1.0,>=0.40.0 in c:\users\dell\anaconda3
\lib\site-packages (from gradio) (0.41.3)
Requirement already satisfied: tomlkit<0.14.0,>=0.12.0 in c:\users\dell\anaconda3
\lib\site-packages (from gradio) (0.12.0)
Requirement already satisfied: typer<1.0,>=0.12 in c:\users\dell\anaconda3\lib\si
te-packages (from gradio) (0.15.1)
Requirement already satisfied: typing-extensions~=4.0 in c:\users\dell\anaconda3
\lib\site-packages (from gradio) (4.11.0)
Requirement already satisfied: uvicorn>=0.14.0 in c:\users\dell\anaconda3\lib\sit
e-packages (from gradio) (0.34.0)
Requirement already satisfied: fsspec in c:\users\dell\anaconda3\lib\site-package
s (from gradio-client==1.6.0->gradio) (2024.3.1)
Requirement already satisfied: websockets<15.0,>=10.0 in c:\users\dell\anaconda3
\lib\site-packages (from gradio-client==1.6.0->gradio) (11.0.3)
Requirement already satisfied: idna>=2.8 in c:\users\dell\anaconda3\lib\site-pack
ages (from anyio<5.0,>=3.0->gradio) (3.7)
```

```
Requirement already satisfied: sniffio>=1.1 in c:\users\dell\anaconda3\lib\site-p
ackages (from anyio<5.0,>=3.0->gradio) (1.3.0)
Requirement already satisfied: certifi in c:\users\dell\anaconda3\lib\site-packag
es (from httpx>=0.24.1->gradio) (2024.7.4)
Requirement already satisfied: httpcore==1.* in c:\users\dell\anaconda3\lib\site-
packages (from httpx>=0.24.1->gradio) (1.0.2)
Requirement already satisfied: h11<0.15,>=0.13 in c:\users\dell\anaconda3\lib\sit
e-packages (from httpcore==1.*->httpx>=0.24.1->gradio) (0.14.0)
Requirement already satisfied: filelock in c:\users\dell\anaconda3\lib\site-packa
ges (from huggingface-hub>=0.25.1->gradio) (3.13.1)
Requirement already satisfied: requests in c:\users\dell\anaconda3\lib\site-packa
ges (from huggingface-hub>=0.25.1->gradio) (2.32.2)
Requirement already satisfied: tqdm>=4.42.1 in c:\users\dell\anaconda3\lib\site-p
ackages (from huggingface-hub>=0.25.1->gradio) (4.66.4)
Requirement already satisfied: python-dateutil>=2.8.2 in c:\users\dell\anaconda3
\lib\site-packages (from pandas<3.0,>=1.0->gradio) (2.9.0.post0)
Requirement already satisfied: pytz>=2020.1 in c:\users\dell\anaconda3\lib\site-p
ackages (from pandas<3.0,>=1.0->gradio) (2024.1)
Requirement already satisfied: tzdata>=2022.7 in c:\users\dell\anaconda3\lib\site
-packages (from pandas<3.0,>=1.0->gradio) (2023.3)
Requirement already satisfied: annotated-types>=0.4.0 in c:\users\dell\anaconda3
\lib\site-packages (from pydantic>=2.0->gradio) (0.6.0)
Requirement already satisfied: pydantic-core==2.14.6 in c:\users\dell\anaconda3\l
ib\site-packages (from pydantic>=2.0->gradio) (2.14.6)
Requirement already satisfied: click>=8.0.0 in c:\users\dell\anaconda3\lib\site-p
ackages (from typer<1.0,>=0.12->gradio) (8.1.8)
Requirement already satisfied: shellingham>=1.3.0 in c:\users\dell\anaconda3\lib
\site-packages (from typer<1.0,>=0.12->gradio) (1.5.4)
Requirement already satisfied: rich>=10.11.0 in c:\users\dell\anaconda3\lib\site-
packages (from typer<1.0,>=0.12->gradio) (13.3.5)
Requirement already satisfied: colorama in c:\users\dell\anaconda3\lib\site-packa
ges (from click>=8.0.0->typer<1.0,>=0.12->gradio) (0.4.6)
Requirement already satisfied: six>=1.5 in c:\users\dell\anaconda3\lib\site-packa
ges (from python-dateutil>=2.8.2->pandas<3.0,>=1.0->gradio) (1.16.0)
Requirement already satisfied: markdown-it-py<3.0.0,>=2.2.0 in c:\users\dell\anac
onda3\lib\site-packages (from rich>=10.11.0->typer<1.0,>=0.12->gradio) (2.2.0)
Requirement already satisfied: pygments<3.0.0,>=2.13.0 in c:\users\dell\anaconda3
\lib\site-packages (from rich>=10.11.0->typer<1.0,>=0.12->gradio) (2.15.1)
Requirement already satisfied: charset-normalizer<4,>=2 in c:\users\dell\anaconda
3\lib\site-packages (from requests->huggingface-hub>=0.25.1->gradio) (2.0.4)
Requirement already satisfied: urllib3<3,>=1.21.1 in c:\users\dell\anaconda3\lib
\site-packages (from requests->huggingface-hub>=0.25.1->gradio) (2.2.2)
Requirement already satisfied: mdurl~=0.1 in c:\users\dell\anaconda3\lib\site-pac
kages (from markdown-it-py<3.0.0,>=2.2.0->rich>=10.11.0->typer<1.0,>=0.12->gradi
o) (0.1.0)
Note: you may need to restart the kernel to use updated packages.
```

In [192…

```python
import gradio as gr
import numpy as np
```

In [193…

```python
def predict(driver_age, driver_experience, previous_accidents, annual_mileage, c
    input_data = np.array([driver_age, driver_experience, previous_accidents, an
    input_data = scaler.transform(input_data)
    prediction = model.predict(input_data)
    return prediction[0]
```

In [217…

```python
iface = gr.Interface(
    fn=predict,
    inputs=[
        gr.Number(label="Driver Age"),
        gr.Number(label="Driver Experience"),
        gr.Number(label="Previous Accidents"),
        gr.Number(label="Annual Mileage"),
        gr.Number(label="Car Manufacturing Year"),
        gr.Number(label="Car Age")
    ],
    outputs= predict(inputs)
)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[217], line 11
      1 iface = gr.Interface(
      2     fn=predict,
      3     inputs=[
      4         gr.Number(label="Driver Age"),
      5         gr.Number(label="Driver Experience"),
      6         gr.Number(label="Previous Accidents"),
      7         gr.Number(label="Annual Mileage"),
      8         gr.Number(label="Car Manufacturing Year"),
      9         gr.Number(label="Car Age")
     10     ],
---> 11     outputs= predict(inputs)
     12 )

NameError: name 'inputs' is not defined
```

In [ ]:
```
iface.launch()
```

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]: