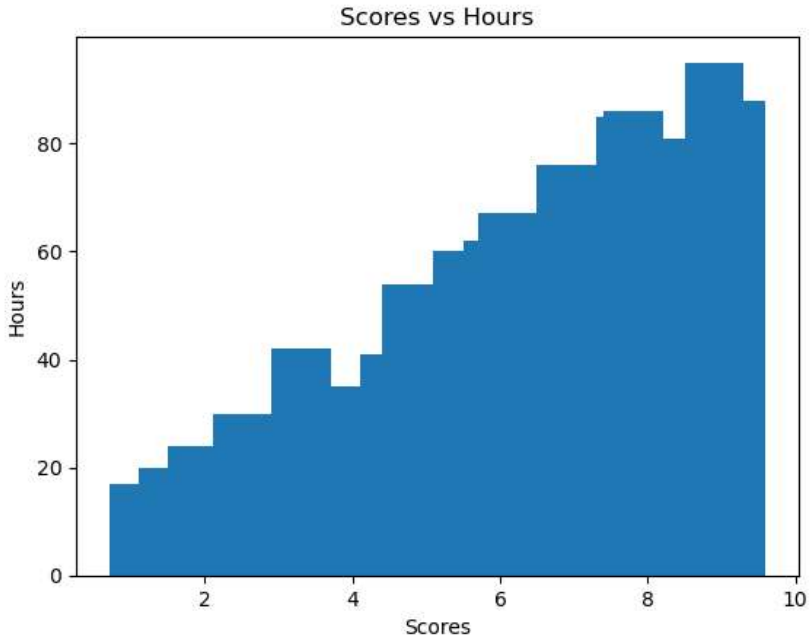
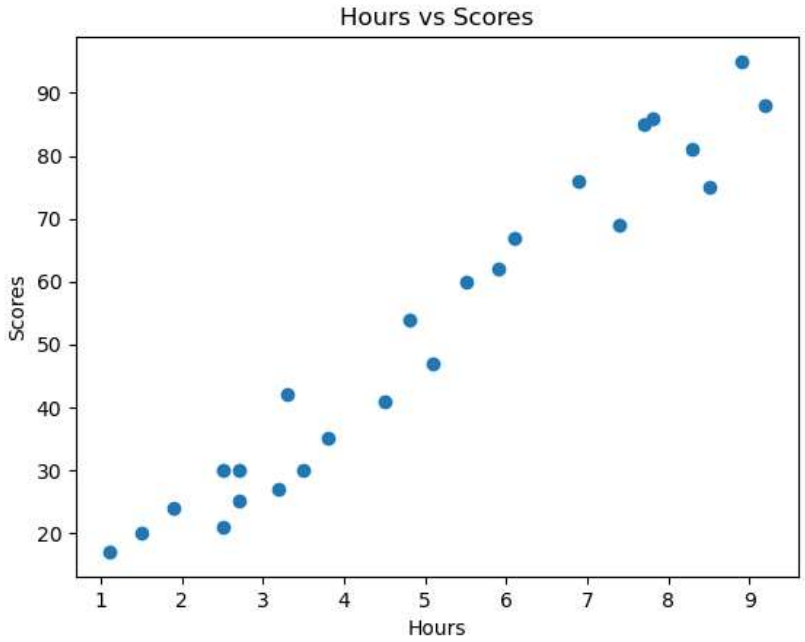
In [2]:

```
 import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import accuracy_score,mean_absolute_error ,mean_squared_error
from sklearn.preprocessing import MinMaxScaler,StandardScaler
from sklearn.compose import ColumnTransformer
from sklearn.ensemble import RandomForestRegressor
import joblib
import matplotlib.pyplot as plt
student_df = pd.read_csv("student_scores.csv")
student_df
scores = student_df["Scores"]
hours = student_df["Hours"]
#bar plot
plt.bar( hours,scores)
#labels and title
plt.xlabel('Scores')
plt.ylabel('Hours')
plt.title('Scores vs Hours')
# show the plot
plt.show()

```



In [3]:
```
scores = student_df["Scores"]
hours = student_df["Hours"]
# Create the bar plot
plt.scatter(hours,scores);
plt.xlabel("Hours")
plt.ylabel("Scores")
plt.title("Hours vs Scores")

```

Out[3]: Text(0.5, 1.0, 'Hours vs Scores')



In [9]:
```
x = np.array(student_df["Hours"]).reshape(-1,1)
y = student_df["Scores"]
# Splitting the Dara
x_train ,x_test ,y_train ,y_test = train_test_split(x,y,test_size=0.3, random_state=41)
regressor= LinearRegression()
regressor.fit(x_train ,y_train)
```

Out[9]:
```
LinearRegression
LinearRegression()
```

In [10]:
```
regressor.score(x_test,y_test)
```

Out[10]: 0.9621346134566173

In [21]:
```
joblib.dump(model,"StudentScore.pkl")
```

Out[21]: ['StudentScore.pkl']

In [22]:
```
model = joblib.load("StudentScore.pkl")
```

In [24]:
```
y_preds = model.predict(x_test)
pd.DataFrame({"y_test": y_test,"y_pred":y_preds})

```

Out[24]:

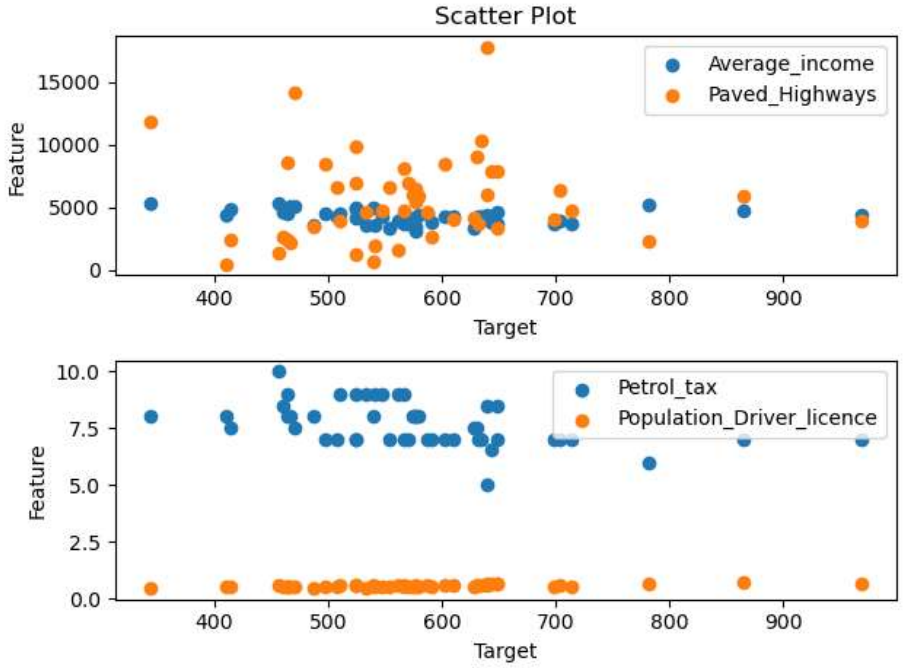
	y_test	y_pred
5	20	14.990287
19	69	74.884076
14	17	10.929691
10	85	77.929523
7	60	55.596245
8	81	84.020416
17	24	19.050883
11	62	59.656841

In [26]:
```
mean_square_error = mean_squared_error(y_test,y_preds)
print(f"mean squared error is {mean_square_error}")
maean_absolute_error = mean_absolute_error(y_test,y_preds)
print(f"mean absolute error is {maean_absolute_error}")

```

```
mean squared error is 25.63250010321114
mean absolute error is 4.843877805244228
```

Task#2

In [28]:
```
Petrol_df = pd.read_csv("petrol_consumption.csv")
Petrol_df.head()
target = Petrol_df['Petrol_Consumption']
Average_income = Petrol_df['Average_income']
Paved_Highways = Petrol_df['Paved_Highways']
Petrol_tax = Petrol_df['Petrol_tax']
Population_Driver_licence = Petrol_df['Population_Driver_licence(%)']
plt.subplot(2, 1, 1)
plt.scatter(target, Average_income, label='Average_income')
plt.scatter(target, Paved_Highways, label='Paved_Highways')
plt.xlabel('Target')
plt.ylabel('Feature')
plt.title('Scatter Plot')
plt.legend()
plt.subplot(2, 1, 2)
plt.scatter(target, Petrol_tax, label='Petrol_tax')
plt.scatter(target, Population_Driver_licence, label='Population_Driver_licence')
plt.xlabel('Target')
plt.ylabel('Feature')
plt.legend()
plt.tight_layout() # Adjusts the spacing between subplots
plt.show()

```



In [31]:
```
minmax =MinMaxScaler()
tranform_colums=["Petrol_tax","Average_income","Paved_Highways","Population_Driver_licence(%)"]
transformer =ColumnTransformer([("minmax",minmax,tranform_colums)])
data = transformer.fit_transform(Petrol_df)
data = pd.DataFrame(data,columns=["Petrol_tax","Average_income","Paved_Highways","Population_Driver_licence(%)"]
data.head()

```

Out[31]:

	Petrol_tax	Average_income	Paved_Highways	Population_Driver_licence(%)
0	0.8	0.222905	0.089044	0.271062
1	0.8	0.451514	0.047202	0.443223
2	0.8	0.351909	0.066567	0.472527
3	0.5	0.792892	0.110656	0.285714
4	0.6	0.586222	0.000000	0.340659

In [32]:
```
x = data
y = Petrol_df["Petrol_Consumption"]
```

In [33]:
```
x_train ,x_test ,y_train ,y_test = train_test_split(x,y,test_size=0.3 ,random_state=52)
model1 = LinearRegression()
model1.fit(x_train ,y_train)
```

Out[33]:
```
LinearRegression
LinearRegression()
```

In [34]:
```
model1.score(x_test,y_test)
```

Out[34]:
```
0.7812556015921516
```

In [37]:
```
joblib.dump(model1,"PetrolComsumption_a.pkl")
mod1 = joblib.load("PetrolComsumption_a.pkl")
y_pred = mod1.predict(x_test)
pd.DataFrame({"y_test": y_test,"y_pred":y_pred})
```

Out[37]:

	y_test	y_pred
8	464	498.887986
3	414	488.882860
25	566	549.211935
44	782	675.788092
18	865	745.446171
26	577	597.792995
6	344	337.080999
9	498	547.265145
34	487	520.150864
17	714	594.451750
1	524	561.120649
30	571	570.485001
0	541	540.348735
38	648	714.693951
21	540	559.453224

In [38]:
```
mean_s_error = mean_squared_error(y_test, y_pred)
print(f"Mean Squared Error: {mean_s_error}")
mean_a_error = mean_absolute_error(y_test, y_pred)
print(f"Mean Absolute Error: {mean_a_error}")
```

```
Mean Squared Error: 3812.248209532195
Mean Absolute Error: 47.09566602370284
```

In [ ]:
```

```