Code:

Q 5.1: Creating a dataset

In [ ]:
```
import pandas as pd
import numpy as np

data={
    'Name': ['Sajeeb','Alamin','Naima','Riaz','Badsa','Badsa Wife','Puti','Montri','Gopal','Kobi'],
    'Age': [34,56,76,45,87,46,46,76,23,45],
    'Gender': ['M','M','F','M','M','M','F','F','M','M'],
    'Marks': [87,88,76,66,56,78,76,78,90,56]
}
df=pd.DataFrame(data)

x = df.to_csv('information.csv', index=False)
```

Q 5.1: Loading a dataset

In [ ]:
```
data=pd.read_csv('information.csv')
print(data)
```

```
        Name  Age Gender  Marks
0     Sajeeb   34      M     87
1     Alamin   56      M     88
2      Naima   76      F     76
3       Riaz   45      M     66
4      Badsa   87      M     56
5  Badsa Wife   46      M     78
6       Puti   46      F     76
7     Montri   76      F     78
8      Gopal   23      M     90
9       Kobi   45      M     56
```

Q 5.2: Finding Mean, Median, Mode, Variance and Standard Deviation

In [ ]:
```
mean = np.mean(data['Marks'])
median = np.median(data['Marks'])
mode_value = data['Marks'].mode()[0]
variance = np.var(data['Marks'])
variance = round(variance,2)
std_dev = np.std(data['Marks'])
std_dev= round(std_dev,2)
# # Print the results
print('Mean:', mean)
print('Median:', median)
print('Mode:', mode_value)
print('Variance:', variance)
print('Standard Deviation:', std_dev)
```

```
Mean: 75.1
Median: 77.0
Mode: 56
Variance: 136.09
Standard Deviation: 11.67
```

Code:

In [ ]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
df=pd.read_csv('1.car driving risk analysis.csv')
df.head()
```

Out[ ]:

	speed	risk
0	200	95
1	90	20
2	300	98
3	110	60
4	240	72

In [ ]:
```
x=df[['speed']]
y=df[['risk']]
x.head()
```

Out[ ]:

	speed
0	200
1	90
2	300
3	110
4	240

In [ ]:
```
y.head()
```

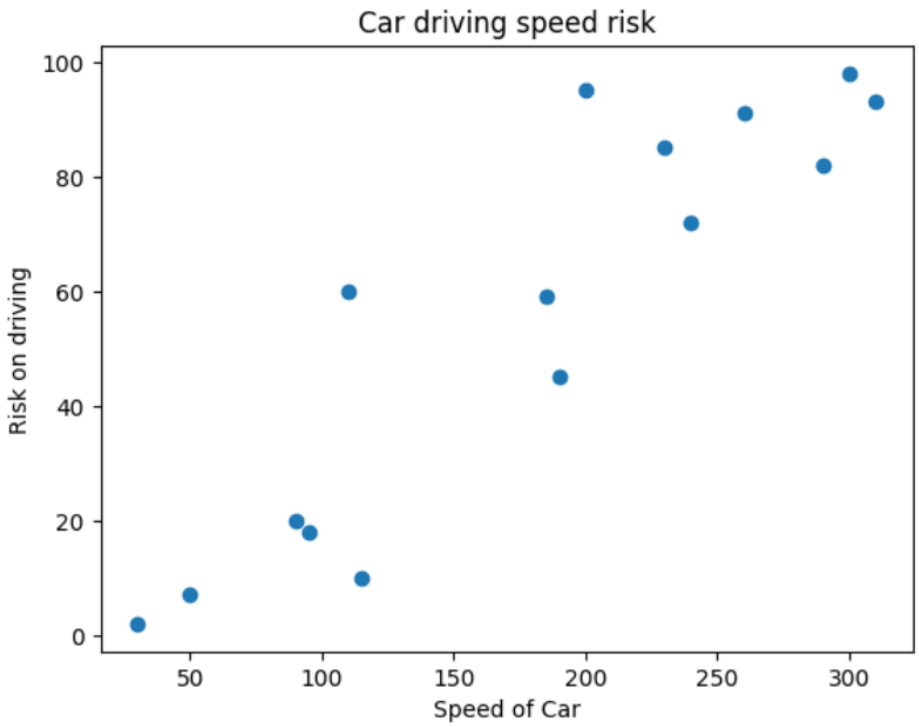
Out[ ]:

	risk
0	95
1	20
2	98
3	60
4	72

Visualization

In [ ]:
```
plt.scatter(x,y)
plt.xlabel('Speed of Car')
plt.ylabel('Risk on driving')
plt.title('Car driving speed risk')
```

Out[ ]:
```
Text(0.5, 1.0, 'Car driving speed risk')
```



60% For training and 40% for test

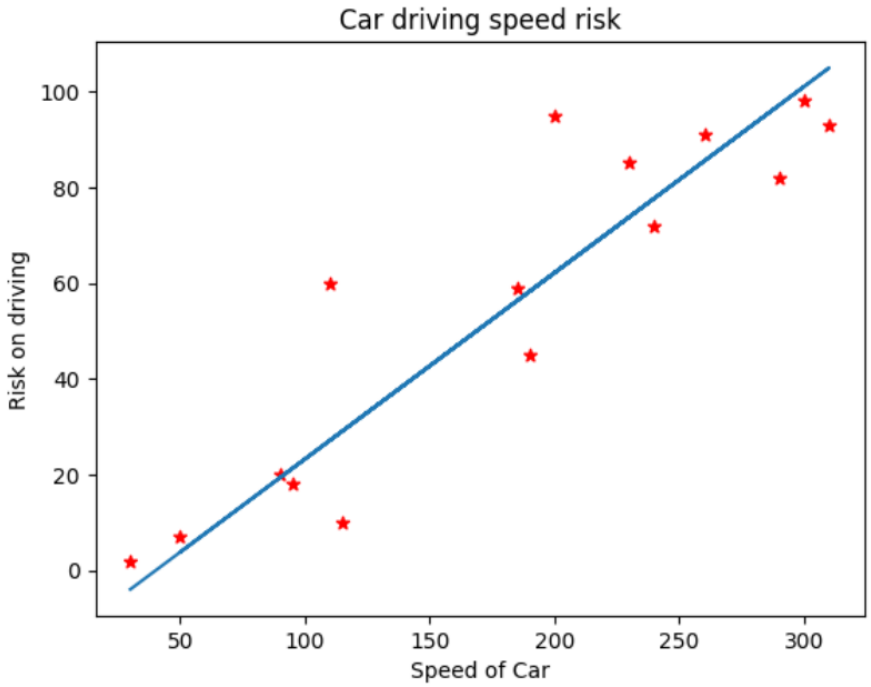
In [ ]:
```
from sklearn.model_selection import train_test_split
xtrain,xtest,ytrain,ytest = train_test_split(x,y,test_size=.40,random_state=1)
```

In [ ]:
```
from sklearn.linear_model import LinearRegression
model=LinearRegression()
model.fit(xtrain,ytrain)
```

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

In [ ]:
```
plt.scatter(x,y,marker='*',color='red')
plt.xlabel('Speed of Car')
plt.ylabel('Risk on driving')
plt.title('Car driving speed risk')
plt.plot(x,model.predict(x))
```

Out[ ]:
```
[<matplotlib.lines.Line2D at 0x16f1487ac10>]
```



In [ ]:
```
coff = model.coef_
print("Coefficient: ", coff)
intercept = model.intercept_
print("intercept: ", intercept)
```

```
Coefficient:  [[0.38891318]]
intercept:  [-15.62743727]
```

In [ ]:
```
y_pred = model.predict(xtest)  #compare with ytest
y_pred
```

Out[ ]:
```
array([[ 27.15301215],
       [ 73.82259334],
       [  3.81822156],
       [101.04651569],
       [ 97.15738393],
       [ 77.7117251 ]])
```

User input speed

In [ ]:
```
risk = model.predict([[150]])
print("When speed 150, then risk is: ", risk[0,0])
```

```
When speed 150, then risk is:  42.709539214829064
C:\Users\sajee\AppData\Local\Packages\PythonSoftwareFoundation.Python.
X does not have valid feature names, but LinearRegression was fitted w
  warnings.warn(
```

In [ ]:
```
print("Model Score is: ")
model.score(xtest,ytest)
```

```
Model Score is:
```

Out[ ]:
```
0.7133824900141749
```

Code:

In [ ]:
```
import pandas as pd
import numpy as np
data=pd.read_csv('data.csv')
data
```

Out[ ]:

	sky	air temp	humidity	wind	water	forecast	enjoy sport
0	sunny	warm	normal	strong	warm	same	yes
1	sunny	warm	high	strong	warm	same	yes
2	rainy	cold	high	strong	warm	change	no
3	sunny	warm	high	strong	cool	change	yes

In [ ]:
```
# leave the last column
concept=np.array(data)[:,:-1]
# only access the last column
target=np.array(data)[:,-1]
print(concept)
print(target)
```

```
[['sunny' 'warm' 'normal' 'strong' 'warm' 'same']
 ['sunny' 'warm' 'high' 'strong' 'warm' 'same']
 ['rainy' 'cold' 'high' 'strong' 'warm' 'change']
 ['sunny' 'warm' 'high' 'strong' 'cool' 'change']]
['yes' 'yes' 'no' 'yes']
```

In [ ]:
```
def train(concept,target):
    for i,value in enumerate(target):
        if value.lower()=='yes':
            specific_h=concept[i].copy()
            break
    for i,value in enumerate(concept):
        if target[i].lower()=='yes':
            for j in range(len(specific_h)):
                if value[j]!=specific_h[j]:
                    specific_h[j]='?'
                else:
                    pass
    return specific_h
```

In [ ]:
```
result=train(concept,target)
print(result)
```

```
['sunny' 'warm' '?' 'strong' '?' '?']
```

In [ ]:
```
day=input("Enter 6 word to check:")
day=day.split()
check=True
```

Enter 6 word to check: sunny warm normal strong warm same

In [ ]:
```
for i in range(len(result)):
    if result[i]=='?'or result[i]==day[i]:
        check=True
    else:
        check=False
        break
if check:
    print("Enjoy sport.")
else:
    print("Not enjoy sport.")
```

```
Enjoy sport.
```

Code:

In [ ]:
```
import numpy as np
import pandas as pd
```

In [ ]:
```
df=pd.read_csv('Social_Network_Ads.csv')
print(df.head())
```

```
   Age  EstimatedSalary  Purchased
0   19            19000          0
1   35            20000          0
2   26            43000          0
3   27            57000          0
4   19            76000          0
```

In [ ]:
```
x=df.iloc[:,[0,1]]
x.head()
```

Out[ ]:

	Age	EstimatedSalary
0	19	19000
1	35	20000
2	26	43000
3	27	57000
4	19	76000

In [ ]:
```
y=df.iloc[:,2]
y.head()
```

Out[ ]:
```
0    0
1    0
2    0
3    0
4    0
Name: Purchased, dtype: int64
```

In [ ]:
```
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC
xtrain,xtest,ytrain,ytest = train_test_split(x,y,test_size=.40,random_state=1)
```

In [ ]:
```
print('Training data : ',xtrain.shape)
print('Testing data : ',xtest.shape)
xtrain.head()
```

```
Training data :  (240, 2)
Testing data :  (160, 2)
```

Out[ ]:

	Age	EstimatedSalary
163	35	38000
247	57	122000
378	41	87000
145	24	89000
251	37	52000

In [ ]:
```
ytrain.head()
```

Out[ ]:
```
163    0
247    1
378    1
145    0
251    0
Name: Purchased, dtype: int64
```

In [ ]:
```
model=SVC(gamma='auto')
model.fit(xtrain,ytrain)
```

Out[ ]:
```
SVC(gamma='auto')
```

In [ ]:
```
model.score(xtest,ytest)
```

Out[ ]:
```
0.66875
```