```
import pandas as pd
import numpy as pd
```

```
import pandas as pd
df=pd.read_csv("/content/insurance - insurance.csv")
df
```

	age	sex	bmi	children	smoker	region	charges
0	19	female	27.900	0	yes	southwest	16884.92400
1	18	male	33.770	1	no	southeast	1725.55230
2	28	male	33.000	3	no	southeast	4449.46200
3	33	male	22.705	0	no	northwest	21984.47061
4	32	male	28.880	0	no	northwest	3866.85520
...	...	...	...	...	...	...	...
1333	50	male	30.970	3	no	northwest	10600.54830
1334	18	female	31.920	0	no	northeast	2205.98080
1335	18	female	36.850	0	no	southeast	1629.83350
1336	21	female	25.800	0	no	southwest	2007.94500
1337	61	female	29.070	0	yes	northwest	29141.36030

1338 rows × 7 columns

Next steps: Generate code with df New interactive sheet

```
from sklearn.preprocessing import LabelEncoder
le=LabelEncoder()
```

```
df['sex']=le.fit_transform(df['sex'])
df['smoker']=le.fit_transform(df['smoker'])
df['region']=le.fit_transform(df['region'])
```

```
df.head(2)
```

	age	sex	bmi	children	smoker	region	charges
0	19	0	27.90	0	1	3	16884.9240
1	18	1	33.77	1	0	2	1725.5523

Next steps: Generate code with df New interactive sheet

```
x=df.drop(columns=['charges'])
y=df['charges']
```

```
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_stat
```

```
from sklearn.linear_model import LinearRegression
lr=LinearRegression()
```

```
lr.fit(x_train,y_train)
```

```
▾ LinearRegression  i ?
LinearRegression()
```

```
y_pred=lr.predict(x_test)
```

```
from sklearn.metrics import r2_score
```

```
r2_score(y_test, y_pred)
```

```
0.7623224022563388
```

```
from sklearn.tree import DecisionTreeRegressor
dt=DecisionTreeRegressor()
```

```
dtr=dt.fit(x_train,y_train)
```

```
y_pred=dtr.predict(x_test)
```

```
r2_score(y_test,y_pred)
```

```
0.744569517123006
```

```
from sklearn.linear_model import Ridge
rd=Ridge()
```

```
rd.fit(x_train,y_train)
```

```
▾ Ridge  i ?
Ridge()
```

```
y_pred=rd.predict(x_test)
```

```
r2_score(y_test,y_pred)
```

```
0.7622110770540812
```

```
from sklearn.linear_model import Lasso
ls=Lasso()
```

```
ls.fit(x_train,y_train)
```

```
▾ Lasso ⓘ ?

Lasso()
```

```
y_pred=ls.predict(x_test)
```

```
r2_score(y_test,y_pred)
```

```
0.7623184615600569
```

```
from sklearn.ensemble import RandomForestRegressor
rf=RandomForestRegressor()
```

```
rfr=rf.fit(x_train,y_train)
```

```
y_pred=rfr.predict(x_test)
```

```
 r2_score(y_test,y_pred)
```

```
0.8537950805139781
```

```
import pandas as pd
import numpy as np
```

```
df=pd.read_csv("/content/covid_toy - covid_toy (1).csv")
df
```

	age	gender	fever	cough	city	has_covid
0	60	Male	103.0	Mild	Kolkata	No
1	27	Male	100.0	Mild	Delhi	Yes
2	42	Male	101.0	Mild	Delhi	No
3	31	Female	98.0	Mild	Kolkata	No
4	65	Female	101.0	Mild	Mumbai	No
...	...	...	...	...	...	...
95	12	Female	104.0	Mild	Bangalore	No
96	51	Female	101.0	Strong	Kolkata	Yes
97	20	Female	101.0	Mild	Bangalore	No
98	5	Female	98.0	Strong	Mumbai	No
99	10	Female	98.0	Strong	Kolkata	Yes

100 rows × 6 columns

Next steps: Generate code with df New interactive sheet

```
from sklearn.preprocessing import LabelEncoder
le=LabelEncoder()
```

```
df['city']=le.fit_transform(df['city'])
df['cough']=le.fit_transform(df['cough'])
df['gender']=le.fit_transform(df['gender'])
```

```
x=df.drop(columns=['has_covid'])
y=df['has_covid']
```

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=42,random_state
```

```
from sklearn.linear_model import LogisticRegression
lr=LogisticRegression()
```

```
lr.fit(x_train, y_train)
```

```
▾ LogisticRegression
LogisticRegression()
```

```
from sklearn.metrics import accuracy_score
accuracy_score(y_test, y_pred)
```

```
0.5714285714285714
```

```
from sklearn.tree import DecisionTreeClassifier
dt=DecisionTreeClassifier()
```

```
lr.fit(x_train_imputed, y_train)
```

```
▾ LogisticRegression  i ?
LogisticRegression()
```

```
y_pred = lr.predict(x_test)
```

```
/usr/local/lib/python3.12/dist-packages/sklearn/utils/validation.py:2739: Use
  warnings.warn(
```

```
from sklearn.metrics import accuracy_score
accuracy_score(y_test, y_pred)
```

```
0.5714285714285714
```

```
from sklearn.ensemble import RandomForestClassifier
rf=RandomForestClassifier()
```

```
lr.fit(x_train, y_train)
```

```
▾ LogisticRegression  i ?
LogisticRegression()
```

```
y_pred=lr.predict(x_test)
```

```
from sklearn.metrics import accuracy_score
accuracy_score(y_test, y_pred)
```

```
0.5714285714285714
```