In [1]:
```
import pandas as pd
import warnings
warnings.filterwarnings('ignore')
```

In [2]:
```
df=pd.read_csv("cancer.csv")
df.head()
```

Out[2]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactne
0	842302	M	17.99	10.38	122.80	1001.0	0.11840	
1	842517	M	20.57	17.77	132.90	1326.0	0.08474	
2	84300903	M	19.69	21.25	130.00	1203.0	0.10960	
3	84348301	M	11.42	20.38	77.58	386.1	0.14250	
4	84358402	M	20.29	14.34	135.10	1297.0	0.10030	

5 rows × 33 columns

In [3]:
```
df.shape
```

Out[3]:
```
(569, 33)
```

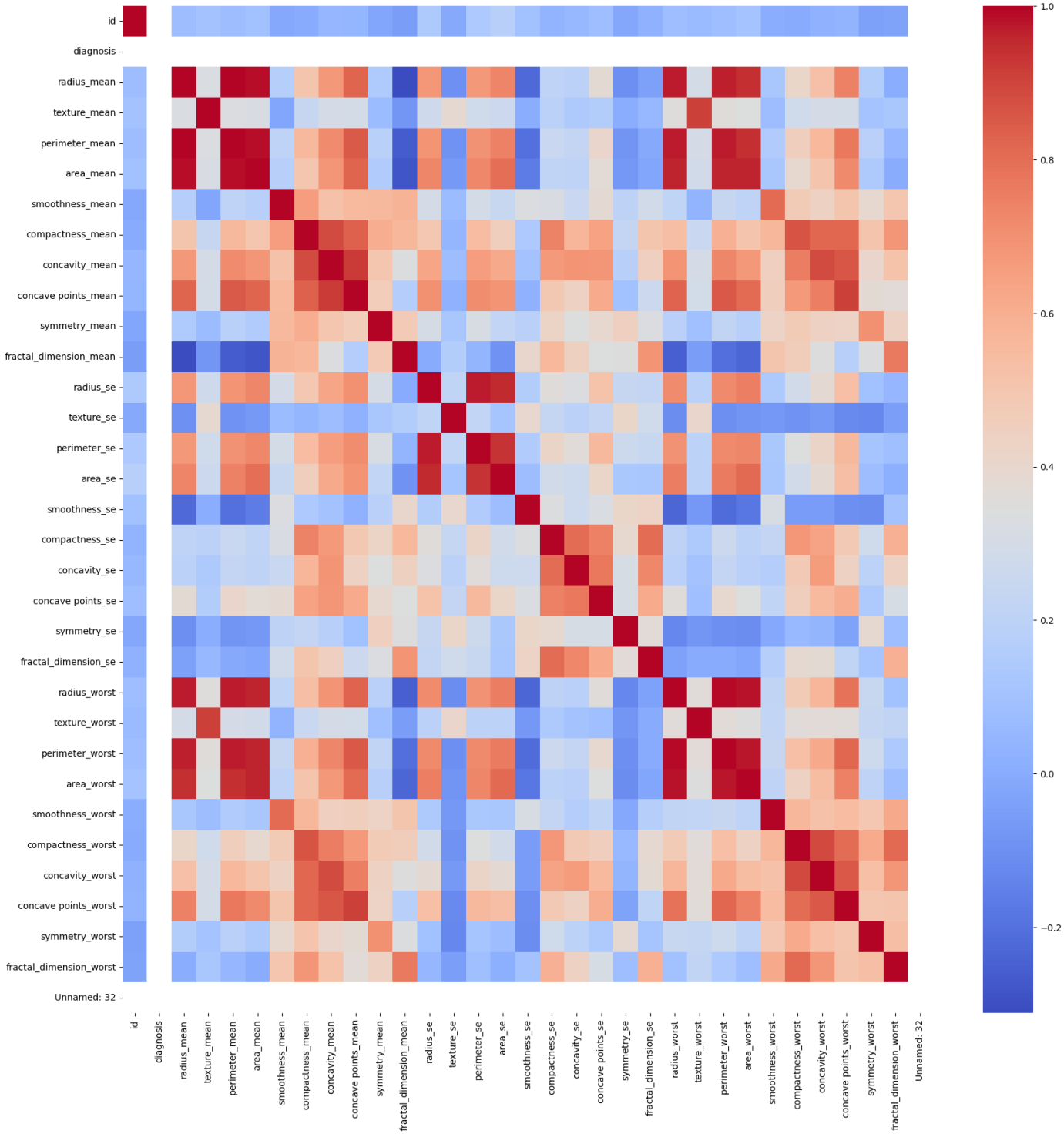
In [4]:
```
df.isnull().sum()
```

Out[4]:
```
id                         0
diagnosis                  0
radius_mean                0
texture_mean               0
perimeter_mean             0
area_mean                  0
smoothness_mean            0
compactness_mean           0
concavity_mean             0
concave points_mean        0
symmetry_mean              0
fractal_dimension_mean     0
radius_se                  0
texture_se                 0
perimeter_se               0
area_se                    0
smoothness_se              0
compactness_se             0
concavity_se               0
concave points_se          0
symmetry_se                0
fractal_dimension_se       0
radius_worst               0
texture_worst              0
perimeter_worst            0
area_worst                 0
smoothness_worst           0
compactness_worst          0
concavity_worst            0
concave points_worst       0
symmetry_worst             0
fractal_dimension_worst    0
Unnamed: 32              569
dtype: int64
```

In [5]:
```
df.diagnosis.value_counts()
```

Out[5]:
```
B    357
M    212
Name: diagnosis, dtype: int64
```

In [32]:
```
import seaborn as sns
import matplotlib.pyplot as plt
sns.heatmap(df.corr(), cmap='coolwarm')
plt.show()
```



In [6]:
```
x=df.iloc[:, 2:32].values
y=df.diagnosis
```

In [7]:
```
df.diagnosis = df.diagnosis.map({'M': 1, 'B': 0})
```

In [8]:
```
from sklearn.preprocessing import StandardScaler  #scaling
Scaler=StandardScaler()
x_Scaled=Scaler.fit_transform(x)
x_Scaled
```

Out[8]:
```
array([[ 1.09706398, -2.07333501,  1.26993369, ...,  2.29607613,
         2.75062224,  1.93701461],
       [ 1.82982061, -0.35363241,  1.68595471, ...,  1.0870843 ,
        -0.24388967,  0.28118999],
       [ 1.57988811,  0.45618695,  1.56650313, ...,  1.95500035,
         1.152255  ,  0.20139121],
       ...,
       [ 0.70228425,  2.0455738 ,  0.67267578, ...,  0.41406869,
        -1.10454895, -0.31840916],
       [ 1.83834103,  2.33645719,  1.98252415, ...,  2.28998549,
         1.91908301,  2.21963528],
       [-1.80840125,  1.22179204, -1.81438851, ..., -1.74506282,
        -0.04813821, -0.75120669]])
```

In [9]:
```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.25,random_state=10)
```

In [10]:
```
x_train.shape
```

Out[10]:
```
(426, 30)
```

In [11]:
```
y_train.shape
```

Out[11]:
```
(426,)
```

In [12]:
```
x_test.shape
```

Out[12]:
```
(143, 30)
```

In [13]:
```
y_test.shape
```

Out[13]:
```
(143,)
```

In [14]:
```
y_train.value_counts()
```

Out[14]:
```
B    266
M    160
Name: diagnosis, dtype: int64
```

In [15]:
```
#using svm method(kernal=rbf)
from sklearn.svm import SVC
model=SVC()
model.fit(x_train,y_train)
```

Out[15]:
```
SVC()
```

In [16]:
```
model.score(x_test,y_test)
```

Out[16]:
```
0.9230769230769231
```

In [17]:
```
#using kernal=linear
from sklearn.svm import SVC
model=SVC(kernel='linear')
model.fit(x_train,y_train)
```

Out[17]:
```
SVC(kernel='linear')
```

In [18]:
```
model.score(x_test,y_test)
```

Out[18]:
```
0.951048951048951
```

In [22]:
```
# Decision Tree
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import cross_val_score
scores=cross_val_score(DecisionTreeClassifier(),x,y,cv=6)
```

In [23]:
```
scores.mean()
```

Out[23]:
```
0.9314109742441209
```

In [24]:
```
#logistic regression
from sklearn.linear_model import LogisticRegression
lr=LogisticRegression()
lr.fit(x_train,y_train)
```

Out[24]:
```
LogisticRegression()
```

In [25]:
```
lr.score(x_test,y_test)
```

Out[25]:
```
0.916083916083916
```

In [26]:
```
#RandomForest classifier
from sklearn.ensemble import RandomForestClassifier
rf=RandomForestClassifier(n_estimators=20)
rf.fit(x_train,y_train)
```

Out[26]:
```
RandomForestClassifier(n_estimators=20)
```

In [27]:
```
rf.score(x_test,y_test)
```

Out[27]:
```
0.9790209790209791
```

In [ ]:
```

```

In [ ]:
```

```