lung

July 23, 2025

[1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report, confusion_matrix,␣
 ↪accuracy_score
```

[2]:
```
df = pd.read_csv("/Users/hashithareddy/Desktop/Lung.csv")
```

[3]:
```
df.head()
```

[3]:
```
   id   age  gender      country diagnosis_date cancer_stage family_history  \
0   1  64.0    Male       Sweden     2016-04-05      Stage I            Yes
1   2  50.0  Female  Netherlands     2023-04-20    Stage III            Yes
2   3  65.0  Female      Hungary     2023-04-05    Stage III            Yes
3   4  51.0  Female      Belgium     2016-02-05      Stage I             No
4   5  37.0    Male   Luxembourg     2023-11-29      Stage I             No

   smoking_status   bmi  cholesterol_level  hypertension  asthma  cirrhosis  \
0  Passive Smoker  29.4                199             0       0          1
1  Passive Smoker  41.2                280             1       1          0
2   Former Smoker  44.0                268             1       1          0
3  Passive Smoker  43.0                241             1       1          0
4  Passive Smoker  19.7                178             0       0          0

   other_cancer treatment_type end_treatment_date  survived
0             0   Chemotherapy         2017-09-10         0
1             0        Surgery         2024-06-17         1
2             0       Combined         2024-04-09         0
3             0   Chemotherapy         2017-04-23         0
4             0       Combined         2025-01-08         0
```

[4]:

```
df.info()
df.isnull().sum()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 890000 entries, 0 to 889999
Data columns (total 17 columns):
 #   Column              Non-Null Count   Dtype
---  ------              --------------   -----
 0   id                  890000 non-null  int64
 1   age                 890000 non-null  float64
 2   gender              890000 non-null  object
 3   country             890000 non-null  object
 4   diagnosis_date      890000 non-null  object
 5   cancer_stage        890000 non-null  object
 6   family_history      890000 non-null  object
 7   smoking_status      890000 non-null  object
 8   bmi                 890000 non-null  float64
 9   cholesterol_level   890000 non-null  int64
 10  hypertension        890000 non-null  int64
 11  asthma              890000 non-null  int64
 12  cirrhosis           890000 non-null  int64
 13  other_cancer        890000 non-null  int64
 14  treatment_type      890000 non-null  object
 15  end_treatment_date  890000 non-null  object
 16  survived            890000 non-null  int64
dtypes: float64(2), int64(7), object(8)
memory usage: 115.4+ MB
```

[4]:

```
id                    0
age                   0
gender                0
country               0
diagnosis_date        0
cancer_stage          0
family_history        0
smoking_status        0
bmi                   0
cholesterol_level     0
hypertension          0
asthma                0
cirrhosis             0
other_cancer          0
treatment_type        0
end_treatment_date    0
survived              0
dtype: int64
```

[5]:

```
df = df.drop(columns=["id", "diagnosis_date", "end_treatment_date"])
```

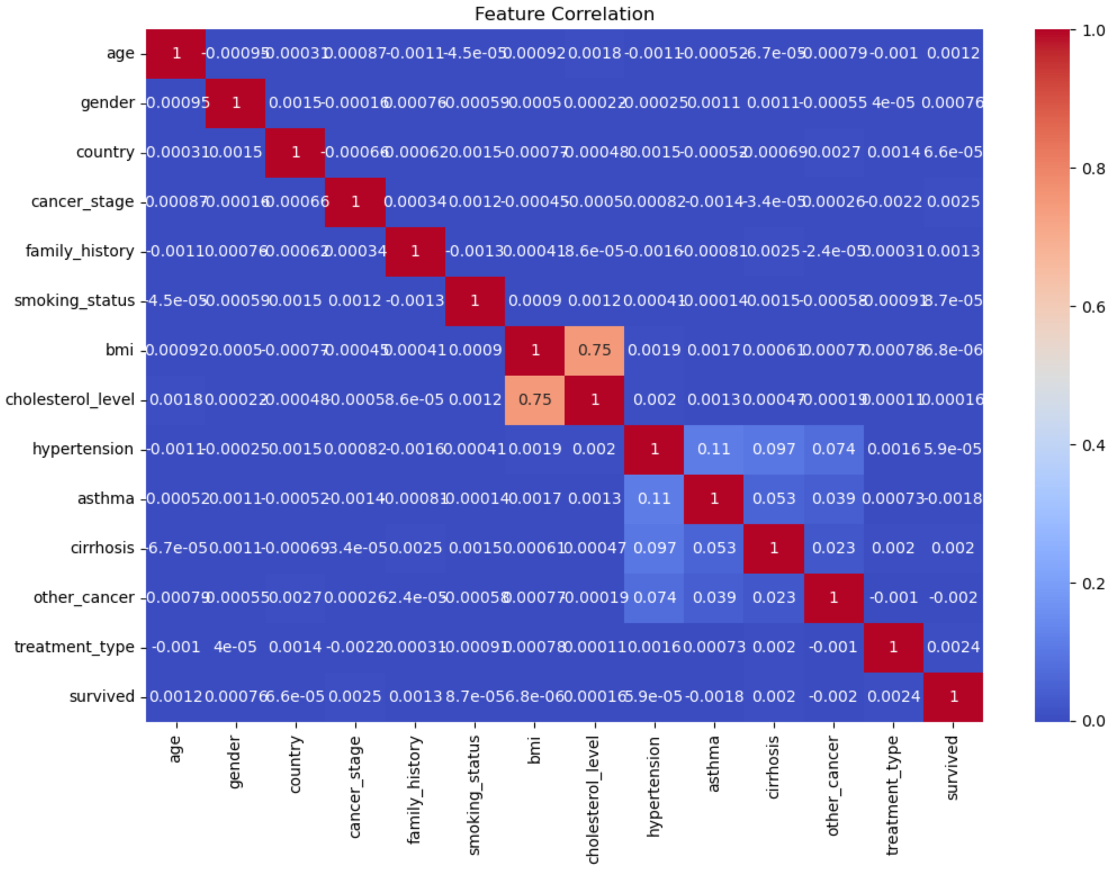
[6]:

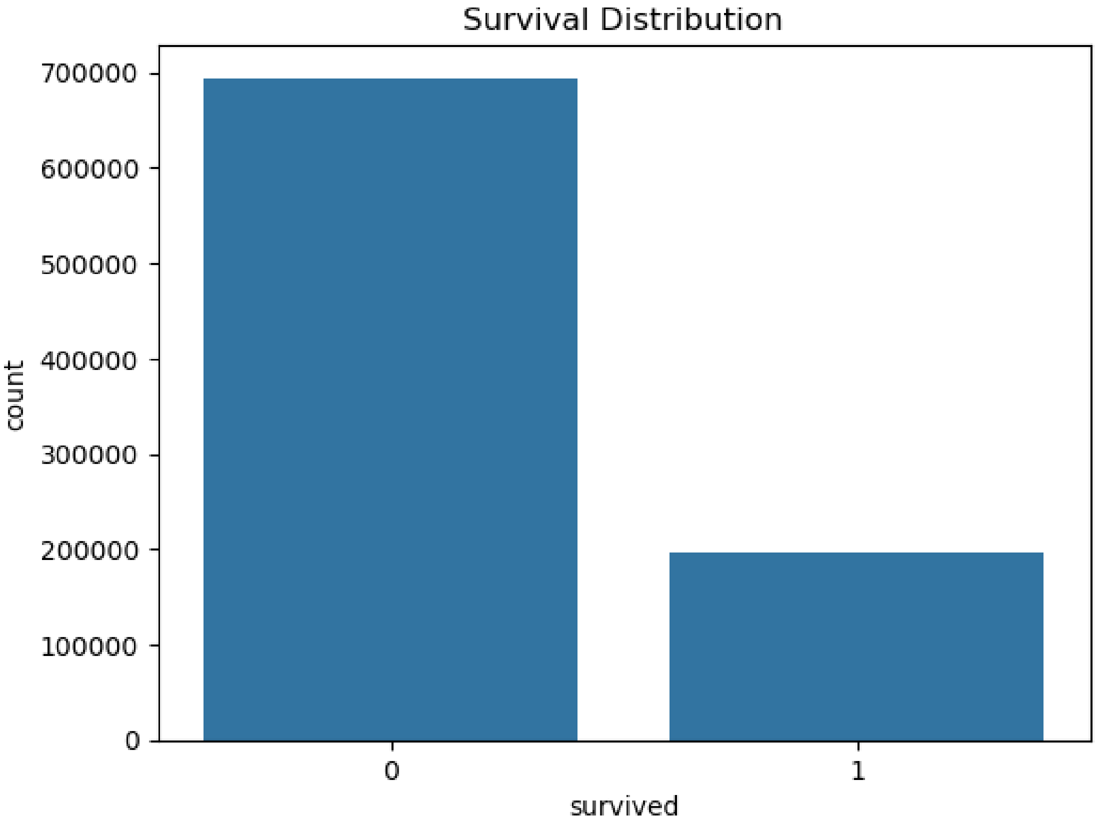
```
label_encoders = {}
for column in df.select_dtypes(include=['object']).columns:
    le = LabelEncoder()
    df[column] = le.fit_transform(df[column])
    label_encoders[column] = le  # store for inverse_transform if needed later
```

[7]:

```
# Correlation heatmap
plt.figure(figsize=(12,8))
sns.heatmap(df.corr(), annot=True, cmap='coolwarm')
plt.title("Feature Correlation")
plt.show()

# Target class distribution
sns.countplot(data=df, x='survived')
plt.title("Survival Distribution")
plt.show()
```





[8]:
```
X = df.drop("survived", axis=1)
y = df["survived"]

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
 ↪random_state=42)
```

[ ]:
```
model = RandomForestClassifier(random_state=42)
model.fit(X_train, y_train)
```

[ ]:
```
y_pred = model.predict(X_test)

print("Accuracy:", accuracy_score(y_test, y_pred))
print("\nClassification Report:\n", classification_report(y_test, y_pred))

# Confusion matrix
sns.heatmap(confusion_matrix(y_test, y_pred), annot=True, fmt='d', cmap='Blues')
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.title("Confusion Matrix")
plt.show()
```

[ ]:

```
import joblib
joblib.dump(model, "lung_cancer_survival_model.pkl")
```

[ ]:

```
pip install xgboost
```

[ ]:

```
from sklearn.linear_model import LogisticRegression
from xgboost import XGBClassifier

models = {
    "Logistic Regression": LogisticRegression(max_iter=1000),
    "Random Forest": RandomForestClassifier(random_state=42),
    "XGBoost": XGBClassifier(use_label_encoder=False, eval_metric='logloss',␣
 ↪random_state=42)
}

for name, clf in models.items():
    clf.fit(X_train, y_train)
    y_pred = clf.predict(X_test)
    print(f"\n{name} Accuracy: {accuracy_score(y_test, y_pred):.4f}")
    print(classification_report(y_test, y_pred))
```

[ ]:

```
# Feature Importance - Random Forest
importances = model.feature_importances_
features = X.columns

# Sort and plot
indices = np.argsort(importances)[::-1]
plt.figure(figsize=(10,6))
sns.barplot(x=importances[indices], y=features[indices])
plt.title("Feature Importance - Random Forest")
plt.show()
```

[35]:

```
import joblib
joblib.dump(model, "lung_cancer_survival_model.pkl")
```

[35]:

```
['lung_cancer_survival_model.pkl']
```

[ ]:

```
```