Code for cancer detection

**%**matplotlib inline

**import** pandas **as** pd

**import** numpy **as** np

**import** matplotlib.pyplot **as** plt

**import** seaborn **as** sns

plt.style.use("seaborn-whitegrid")

add Codeadd Markdown

* **There is also imbalance in data, so let's see how we train and predict with this dataset.**

add Codeadd Markdown

[ ]:



*# Importing our Dataset*

​

cancer\_patient **=** pd.read\_csv("../input/cancer-patientscsv/cancer\_patient.csv")

cancer\_patient.head()

len(cancer\_patient)

[ ]:



cancer\_patient.info();

**Data is already clean, no null values so let's analyse further**

cancer\_patient.describe().T

add Codeadd Markdown

[ ]:



fig **=** plt.figure(figsize **=** (13,8))

sns.heatmap(cancer\_patient.corr(),cmap**=**'coolwarm',annot**=True**);

add Codeadd Markdown

[ ]:



fig, ax **=** plt.subplots()

hist **=** ax.hist(x **=** cancer\_patient["Age"]);

add Codeadd Markdown

[ ]:



*#Required outside of function. This needs to be activated first when plotting in every code block*

fig, ax **=** plt.subplots()

​

*#Count plot*

plot **=** sns.countplot(data **=** cancer\_patient, x**=**'Level', hue**=**'Gender', palette**=**['darkblue','darkred'])

add Codeadd Markdown

[ ]:



cancer\_patient.columns

add Codeadd Markdown

**Cancer found in people age over 50**

add Codeadd Markdown

[ ]:



cancer\_over50 **=** cancer\_patient[cancer\_patient["Age"] **>** 50]

cancer\_over50.head()

add Codeadd Markdown

[ ]:



*# Making Subplots*

fig, ((ax1, ax2), (ax3, ax4)) **=** plt.subplots(nrows **=** 2, ncols**=** 2, figsize**=**(10, 10))

​

*# Adding Data to the plot*

scatter **=** ax1.scatter(x **=** cancer\_over50["Age"], y **=** cancer\_over50["Alcohol use"], cmap **=** "winter")

​

*# For Plot ax1*

ax1.set(title **=** "Age with respect to Alcohol Use",

xlabel **=** "Age",

ylabel **=** "Alcohol Use")

ax1.axhline(cancer\_over50["Alcohol use"].mean(),

linestyle **=** "--");

ax1.set\_xlim([50, 80])

ax1.set\_ylim([0, 8.5])

​

*# For Plot ax2*

scatter **=** ax2.scatter(x **=** cancer\_over50["Age"], y **=** cancer\_over50["Genetic Risk"])

ax2.set(title **=** "Age with respect to Genetic Risk", xlabel **=** "Age", ylabel **=** "Genetic Risk")

ax2.axhline(cancer\_over50["Genetic Risk"].mean(),

linestyle **=** "--");

ax2.set\_xlim([50, 80])

ax2.set\_ylim([0, 7.5])

​

*# For Plot ax3*

scatter **=** ax3.scatter(x **=** cancer\_over50["Age"], y **=** cancer\_over50["Smoking"])

ax3.set(title **=** "Age with respect to Smoking", xlabel **=** "Age", ylabel **=** "Smoking")

ax3.axhline(cancer\_over50["Smoking"].mean(),

linestyle **=** "--");

ax3.set\_xlim([50, 80])

ax3.set\_ylim([0, 8.5])

​

*# For Plot ax4*

scatter **=** ax4.scatter(x **=** cancer\_over50["Gender"], y **=** cancer\_over50["Alcohol use"])

ax4.set(title **=** "Age with respect to Alcohol Use", xlabel **=** "Gender", ylabel **=** "Alcohol Use")

ax4.axhline(cancer\_over50["Alcohol use"].mean(),

linestyle **=** "--");

add Codeadd Markdown

[ ]:



cancer\_over50.head()

add Codeadd Markdown

[ ]:



len(cancer\_patient), len(cancer\_over50)

add Codeadd Markdown

**There are only 134 patients who are Over 50 so we analyse the entire data irresespective of age to achieve fruitfull results later.**

add Codeadd Markdown

[ ]:



cancer\_patient.columns

add Codeadd Markdown

[ ]:



fig, ax **=** plt.subplots()

​

histt **=** ax.hist(x **=** cancer\_patient["Level"], bins **=** 10, color **=**'darkred')

​

ax.set(xlabel **=** "Level", ylabel **=** "Count");

add Codeadd Markdown

[ ]:



cancer\_patient.info()

add Codeadd Markdown

**As we can see Level dtype is not int so first we replace it with numbers then into type int**

add Codeadd Markdown

[ ]:



cancer\_patient["Level"].replace(["Low", "Medium", "High"], ["0", "1", "2"], inplace**=True**)

add Codeadd Markdown

[ ]:



cancer\_patient["Level"] **=** cancer\_patient["Level"].astype(int)

add Codeadd Markdown

[ ]:



cancer\_patient.head().info()

add Codeadd Markdown

**Above entire data is in int**

add Codeadd Markdown

**Plotting with respect to Age and Genetic Risk**

add Codeadd Markdown

[ ]:



fig, ax **=** plt.subplots(figsize **=** (10, 6));

​

scatter **=** ax.scatter(x **=** cancer\_patient["Age"],

y **=** cancer\_patient["Genetic Risk"],

c **=** cancer\_patient["Level"],

cmap **=** "winter")

​

ax.set(xlabel **=** "Age",

ylabel **=** "Genetic Risk");

​

ax.legend(**\***scatter.legend\_elements(), title **=** "Level");

​

ax.axhline(cancer\_patient["Level"].mean(),

linestyle **=** "--");

add Codeadd Markdown

[ ]:



cancer\_patient.plot.kde(figsize **=** (20,8));

add Codeadd Markdown

[ ]:



np.array([cancer\_patient["Gender"][:10]])

add Codeadd Markdown

**Number of Male & Females**

add Codeadd Markdown

[ ]:



male **=** 0

female **=** 0

**for** i **in** cancer\_patient["Gender"]:

**if** i **==** 1:

male **+=** 1

**elif** i **==** 2:

female **+=** 1

f"Number of Male: {male}, Number of females: {female}"

add Codeadd Markdown

[ ]:



*# Make a histogram here*

cancer\_patient\_male **=** cancer\_patient[cancer\_patient["Gender"] **==** 1]

cancer\_patient\_male.head()

add Codeadd Markdown

[ ]:



cancer\_patient\_female **=** cancer\_patient[cancer\_patient["Gender"] **==** 2]

cancer\_patient\_female.head()

add Codeadd Markdown

[ ]:



plt.hist(cancer\_patient\_male);

add Codeadd Markdown

[ ]:



plt.hist(cancer\_patient\_female);

add Codeadd Markdown

[ ]:



cancer\_patient\_male.plot.hist(figsize **=** (15, 50), subplots **=** **True**);

add Codeadd Markdown

[ ]:



cancer\_patient\_female.plot.hist(figsize **=** (15, 50), subplots **=** **True**);

add Codeadd Markdown

[ ]:



fig, ax **=** plt.subplots()

scatter **=** ax.scatter(x **=** cancer\_patient\_male["Alcohol use"], y **=** cancer\_patient\_male["Smoking"])

*# cancer\_patient\_male.plot(x = cancer\_patient\_male["Alcohol use"], y = cancer\_patient\_male["Age"], kind = "scatter");*

add Codeadd Markdown

[ ]:



fig, ax **=** plt.subplots()

scatter **=** ax.scatter(x **=** cancer\_patient\_female["Alcohol use"], y **=** cancer\_patient\_female["Smoking"]);

add Codeadd Markdown

[ ]:



fig, ax **=** plt.subplots()

cancer\_patient\_male.plot(kind **=** "bar", x **=** "Genetic Risk", y **=** "Smoking", ax **=** ax);

add Codeadd Markdown

[ ]:



len(cancer\_patient\_male), len(cancer\_patient\_female)

add Codeadd Markdown

[ ]:



cancer\_patient.head()

add Codeadd Markdown

[ ]:



fig, ax **=** plt.subplots()

cancer\_patient.plot(kind **=** "bar", x **=** "Gender", y **=** "Age", ax **=** ax);

add Codeadd Markdown

[ ]:



fig, ax **=** plt.subplots(figsize **=** (10, 6))

scatter **=** ax.scatter(x **=** cancer\_patient["Age"],

y **=** cancer\_patient["Alcohol use"],

c **=** cancer\_patient["Level"],

cmap **=** "winter")

​

ax.set(xlabel **=** "Age",

ylabel **=** "Alcohol use");

​

ax.legend(**\***scatter.legend\_elements(), title **=** "Level");

​

ax.axhline(cancer\_patient["Level"].mean(),

linestyle **=** "--");

add Codeadd Markdown

[ ]:



fig, ax**=**plt.subplots()*#Required outside of function. This needs to be activated first when plotting in every code block*

plot**=**sns.scatterplot(data**=**cancer\_patient,

x**=**'Alcohol use',

y**=**'Fatigue',

hue**=**'Level',

palette**=**['darkblue','darkred','darkgreen'],

s**=**50,

marker**=**'o')*#Count plot*

add Codeadd Markdown

[ ]:



fig, ax**=**plt.subplots()*#Required outside of function. This needs to be activated first when plotting in every code block*

plot**=**sns.scatterplot(data**=**cancer\_patient,

x**=**'Genetic Risk',

y**=**'Smoking',

hue**=**'Level',

palette**=**['darkblue','darkred','darkgreen'],

s**=**50,

marker**=**'o')*#Count plot*

add Codeadd Markdown

**Our data is analyzed and ready for Model Training and Machine Learning**

add Codeadd Markdown

[ ]:



cancer\_patient.head()

add Codeadd Markdown

[ ]:



cancer\_patient.drop(["Patient Id"], axis **=** 1, inplace**=** **True**)

add Codeadd Markdown

[ ]:



cancer\_patient.head()

add Codeadd Markdown

**Fitting the model/algorithm and use it to make predictions on our data.**

add Codeadd Markdown

**First we use Support Vector Machine Estimator**

add Codeadd Markdown

[ ]:



**from** sklearn **import** svm

**from** sklearn.model\_selection **import** train\_test\_split

​

X **=** cancer\_patient.drop(["Level"], axis **=** 1)

y **=** cancer\_patient["Level"]

​

X\_train, X\_test, y\_train, y\_test **=** train\_test\_split(X, y, test\_size **=** 0.2)

​

sv **=** svm.SVC()

sv.fit(X\_train, y\_train)

sv.score(X\_test, y\_test)

add Codeadd Markdown

[ ]:



y\_preds **=** sv.predict(X\_test)

y\_preds[:10]

add Codeadd Markdown

[ ]:



**from** sklearn.metrics **import** classification\_report, confusion\_matrix, accuracy\_score

print(classification\_report(y\_test, y\_preds))

add Codeadd Markdown

[ ]:



confusion\_matrix(y\_test, y\_preds)

add Codeadd Markdown

[ ]:



accuracy\_score(y\_test, y\_preds)

add Codeadd Markdown

**Checking accuracy with other model**

add Codeadd Markdown

[ ]:



**from** sklearn.neighbors **import** KNeighborsClassifier

**from** sklearn.model\_selection **import** train\_test\_split

​

X **=** cancer\_patient.drop(["Level"], axis **=** 1)

y **=** cancer\_patient["Level"]

​

X\_train, X\_test, y\_train, y\_test **=** train\_test\_split(X, y, test\_size **=** 0.2)

​

knn **=** KNeighborsClassifier()

knn.fit(X\_train, y\_train)

knn.score(X\_test, y\_test)

add Codeadd Markdown

**So the result from KNeighborClassifier has increaed and we can clearly see the change**

add Codeadd Markdown

**Lastly we use RandomForestClassifier**

add Codeadd Markdown

[ ]:



**from** sklearn.ensemble **import** RandomForestRegressor

**from** sklearn.model\_selection **import** train\_test\_split

​

X **=** cancer\_patient.drop(["Level"], axis **=** 1)

y **=** cancer\_patient["Level"]

​

X\_train, X\_test, y\_train, y\_test **=** train\_test\_split(X, y, test\_size **=** 0.2)

​

rfr **=** RandomForestRegressor()

rfr.fit(X\_train, y\_train)

rfr.score(X\_test, y\_test)

add Codeadd Markdown

**Cross val for all the above algorithms to make sure for scores accuracy**

add Codeadd Markdown

[ ]:



**from** sklearn.model\_selection **import** cross\_val\_score

​

crossVal\_sv **=** cross\_val\_score(sv, X, y)

crossVal\_knn **=** cross\_val\_score(knn, X, y)

crossVal\_rfr **=** cross\_val\_score(rfr, X, y)

​

print(f"For SupportVectorMachine: {crossVal\_sv}, \nFor KNeighborClassifier: {crossVal\_knn}, \nFor RandomForestRegressor: {crossVal\_rfr}")

add Codeadd Markdown

**Comparing Score results with Cross Value Score**

add Codeadd Markdown

[



*# For SupportVectorMachine*

​

np.random.seed(42)

​

sv\_single\_score **=** sv.score(X\_test, y\_test)

​

sv\_cross\_val\_score **=** np.mean(cross\_val\_score(sv, X, y))

​

sv\_single\_score, sv\_cross\_val\_score

add Codeadd Markdown

[ ]:



*# For KNeighborClassifier*

​

np.random.seed(42)

​

knn\_single\_score **=** knn.score(X\_test, y\_test)

​

knn\_cross\_val\_score **=** np.mean(cross\_val\_score(knn, X, y))

​

knn\_single\_score, knn\_cross\_val\_score