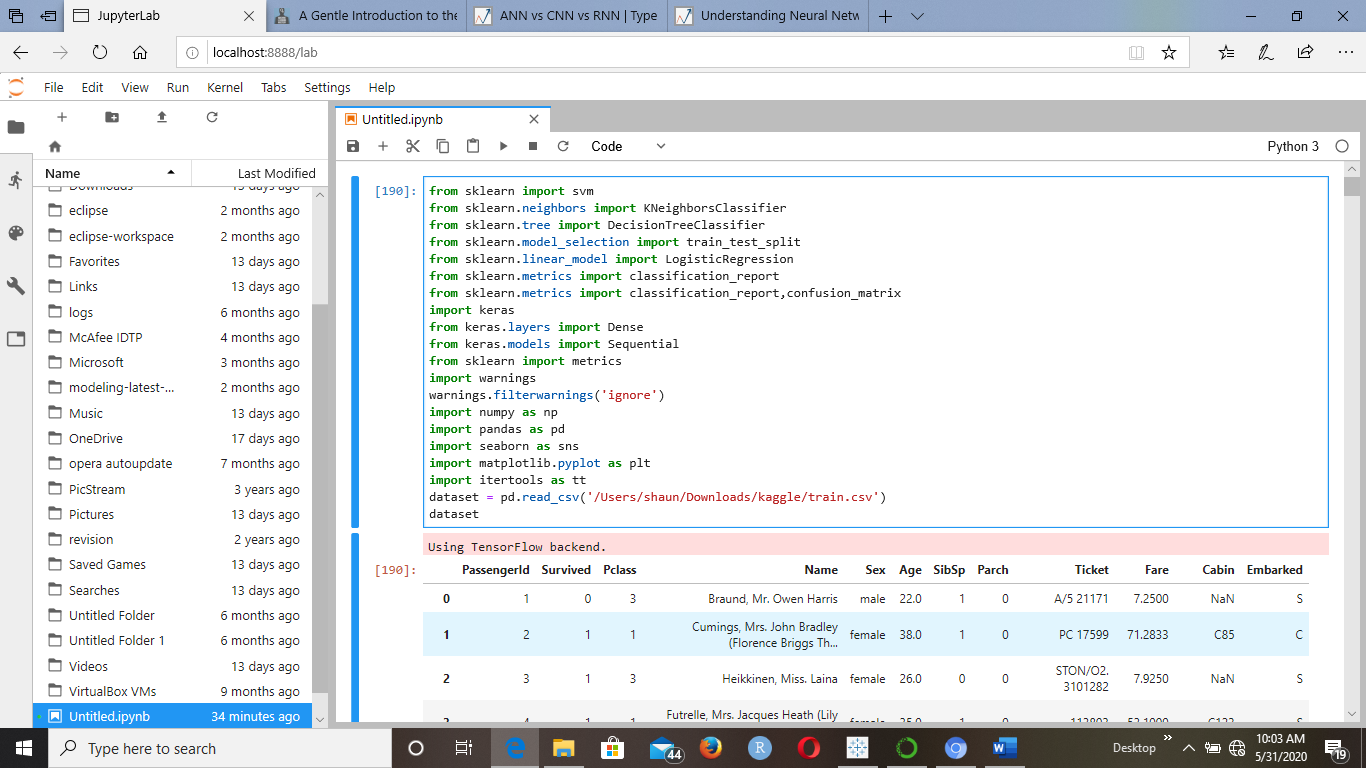
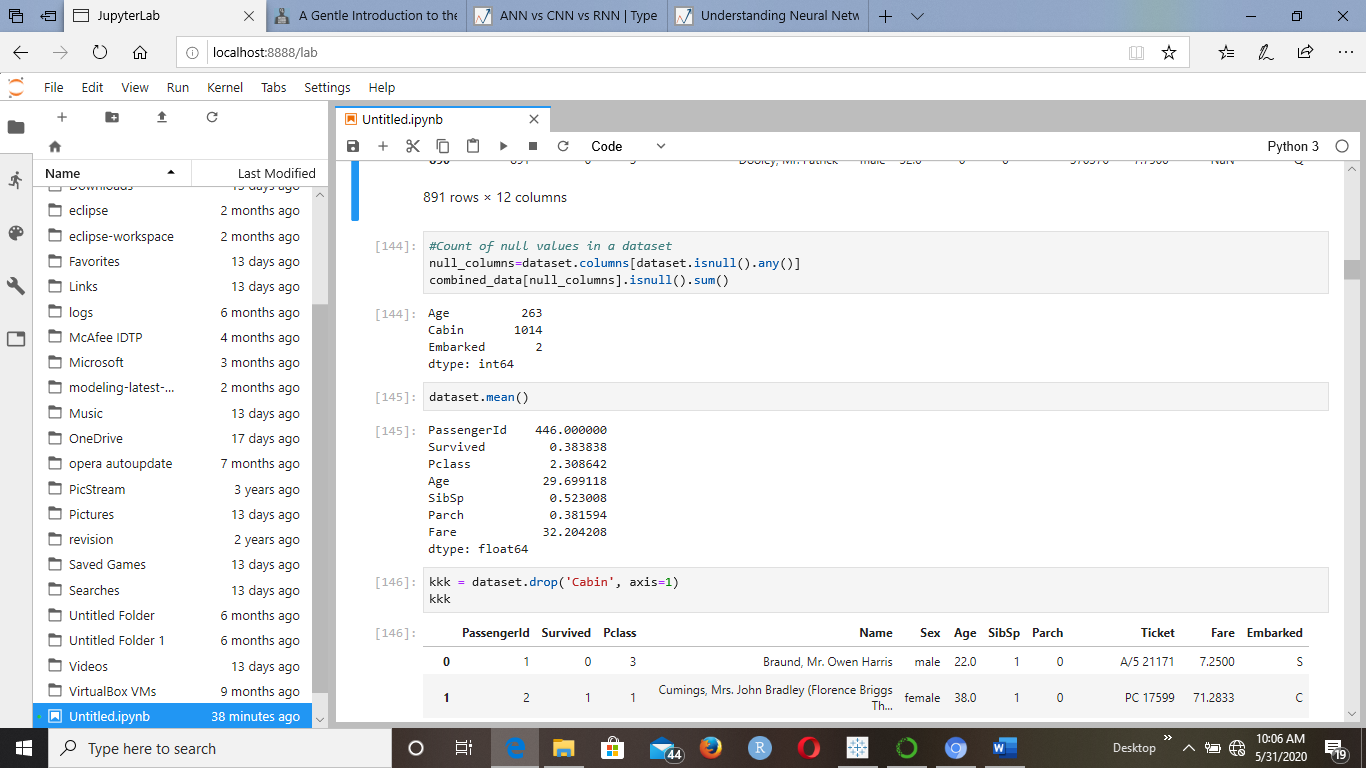
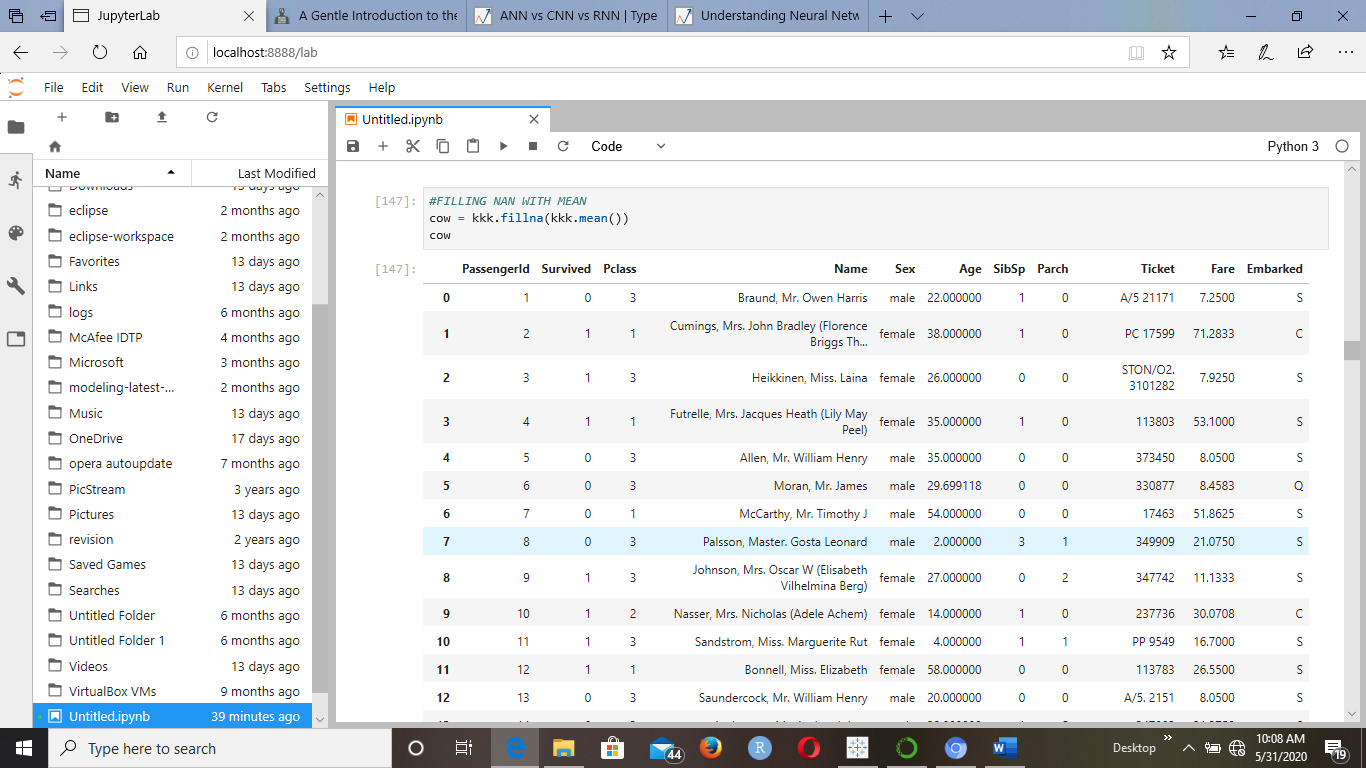
1. IMPORTING DATA TRAIN DATASET



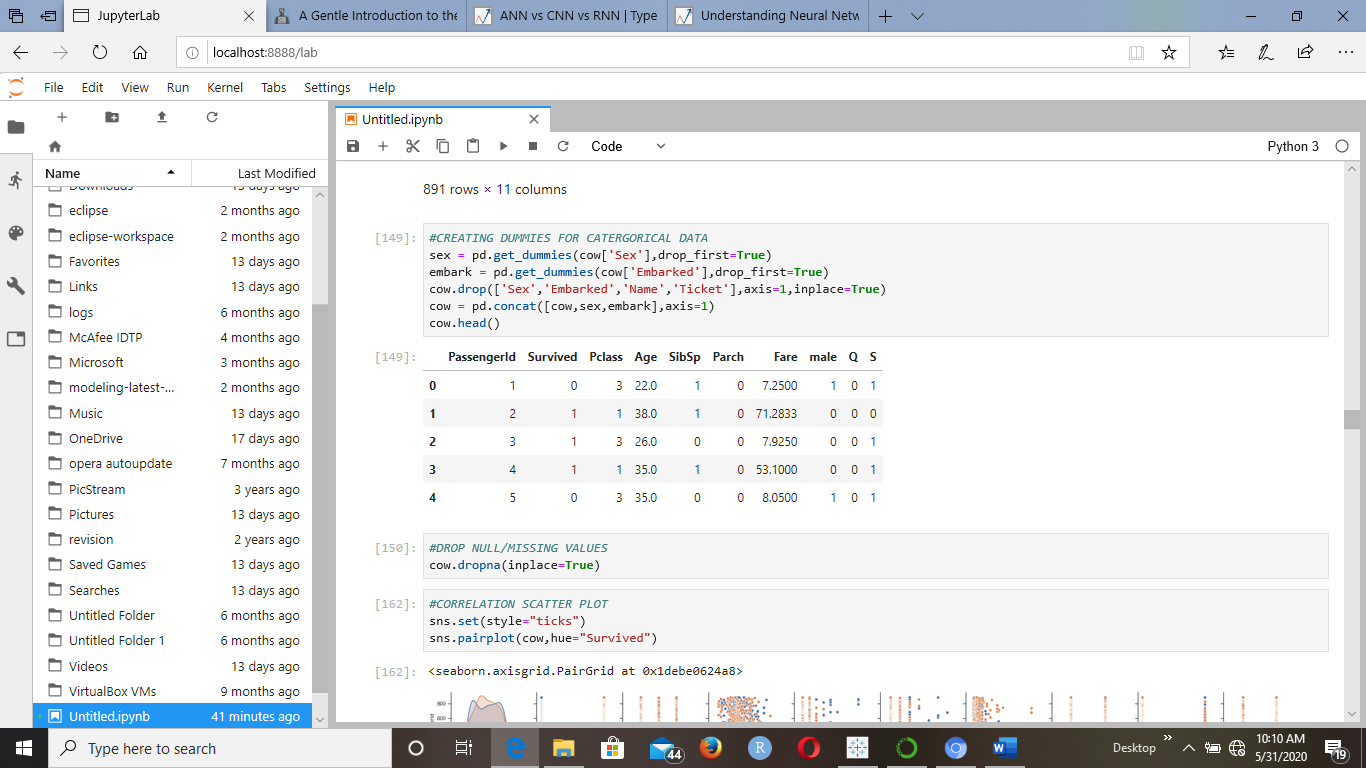
1. Count of null values + dataset mean + dropping the “Cabin” variable



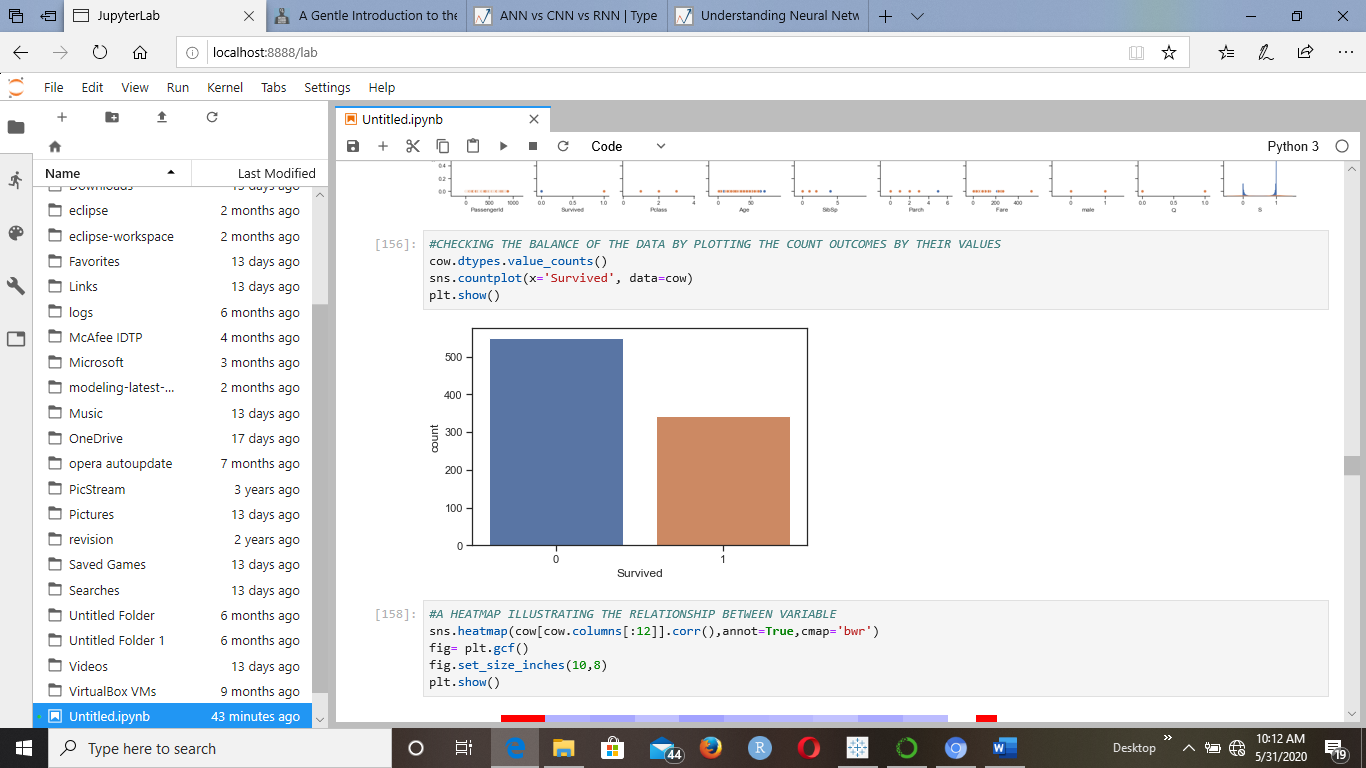
1. Filling Nan with mean



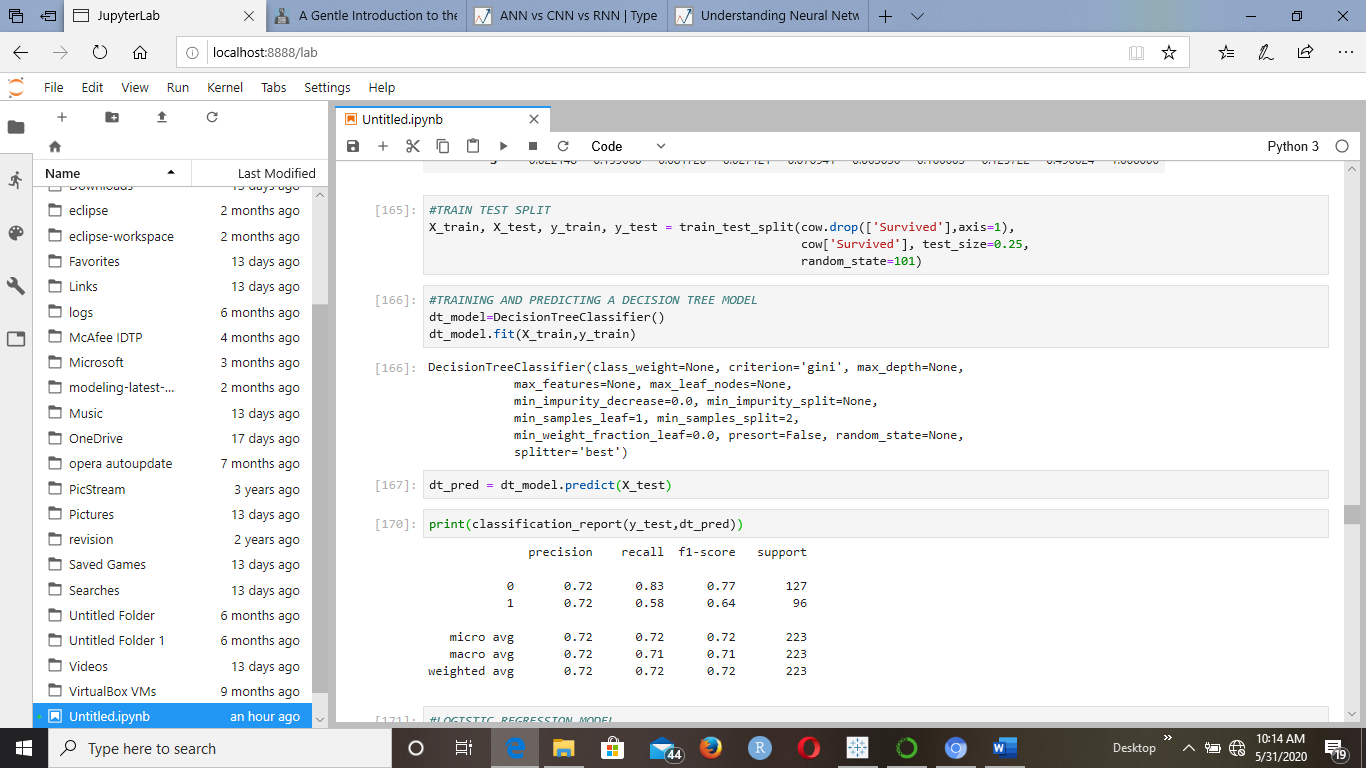
1. Creating dummies for categorical data + dropping null values + correlation scatterplot



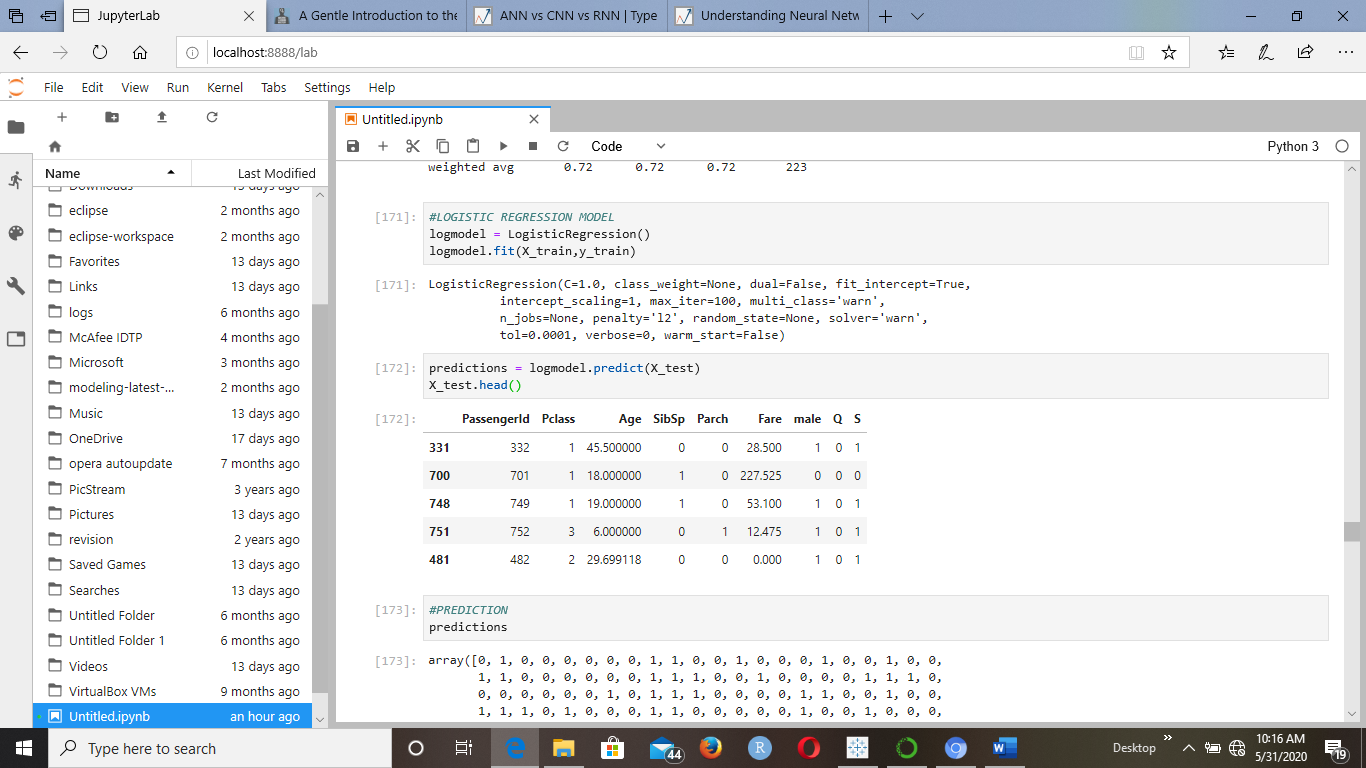
1. Checking the balance of data + heatmap



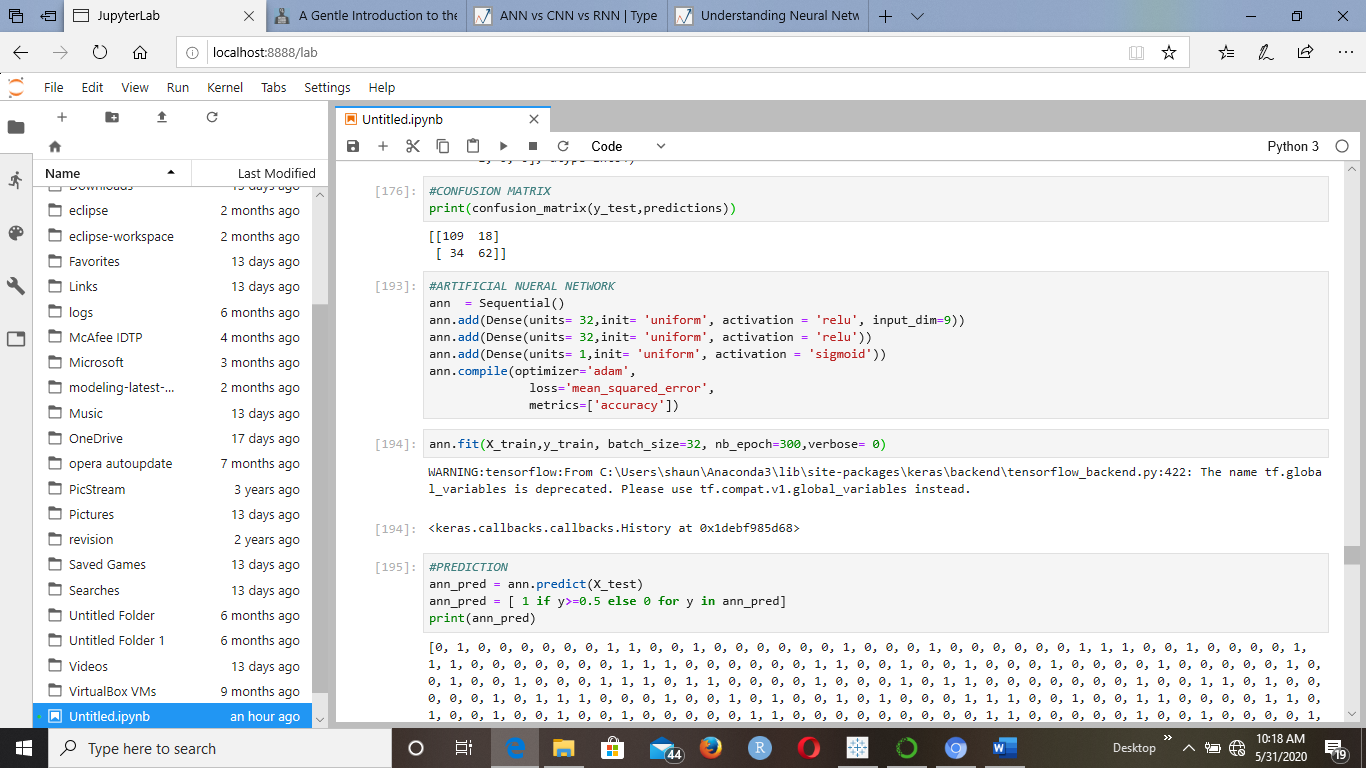
1. Machine learning ( train test split) + Decision Tree + Classification report



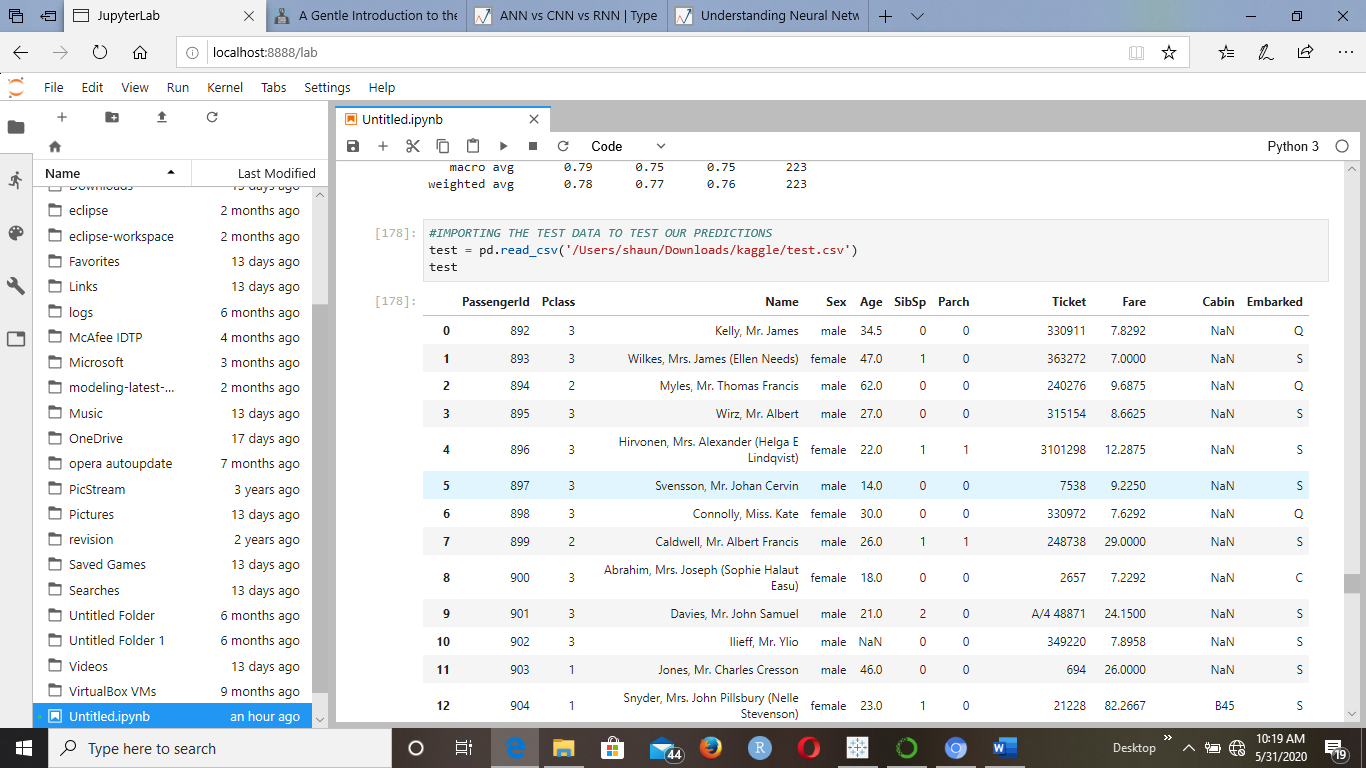
1. Logistic regression modeling, prediction and report



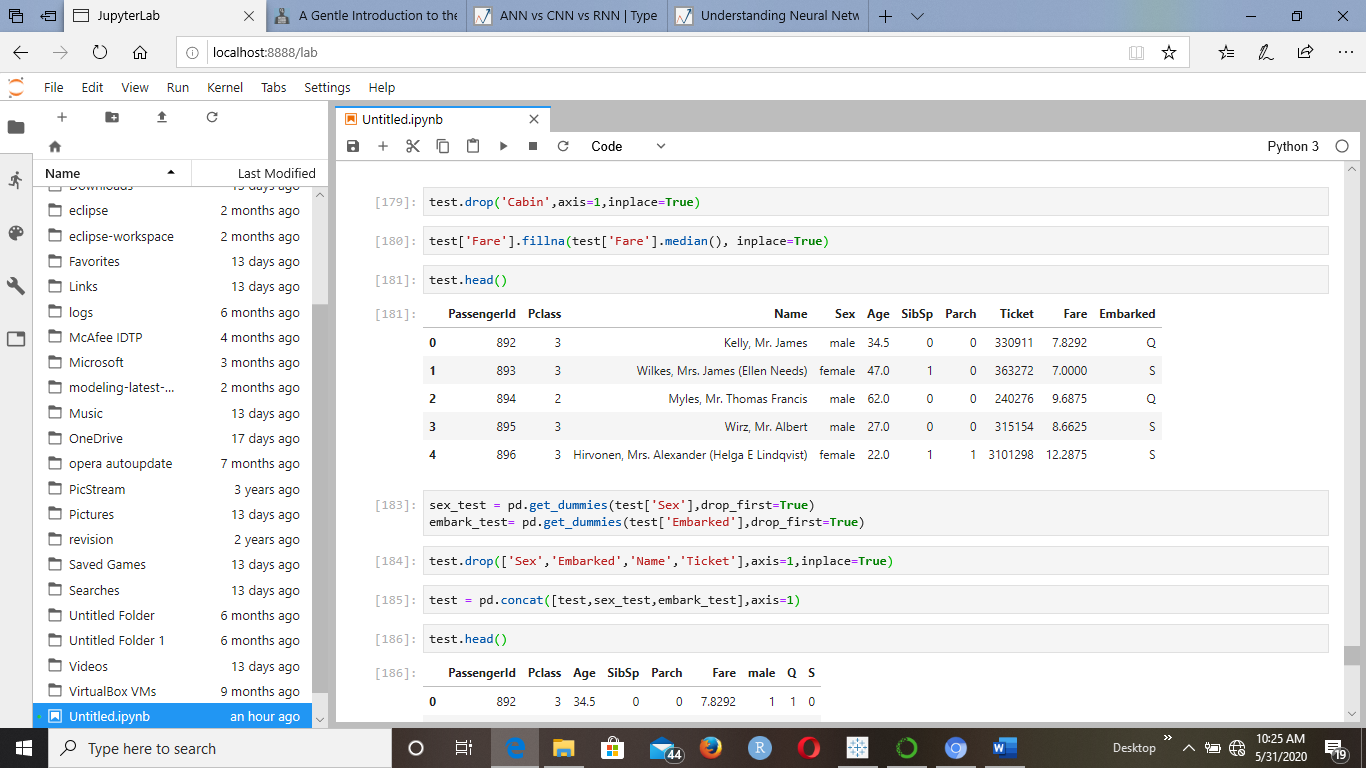
1. Artificial Neural Network modeling, prediction and reporting



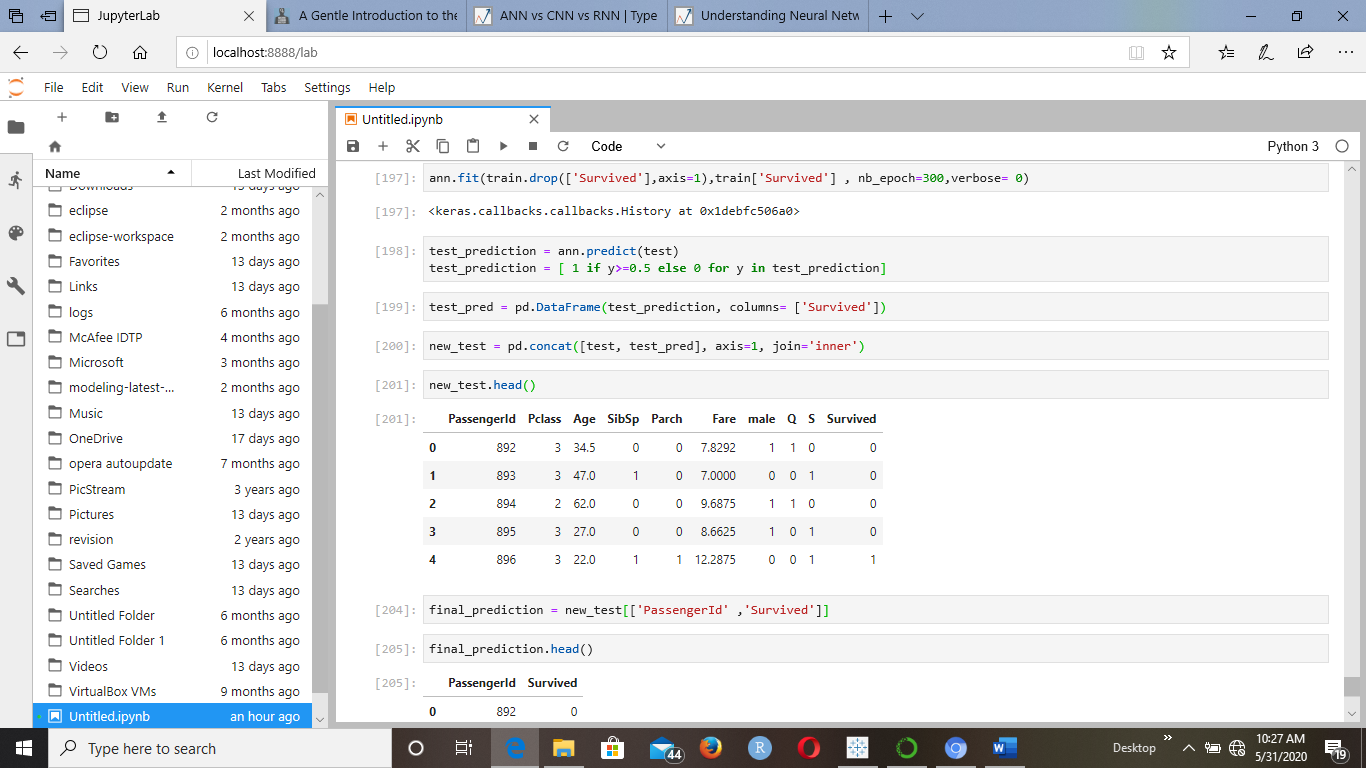
1. Importing the test dataset



1. Dropping the “Cabin” variable + filling in a median + creating dummies + concatenating data



1. Using ANN to predict + concatenating data to show final outcome



**Code snippet**

**from sklearn import svm**

**from sklearn.neighbors import KNeighborsClassifier**

**from sklearn.tree import DecisionTreeClassifier**

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

**from sklearn.linear\_model import LogisticRegression**

**from sklearn.metrics import classification\_report**

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

**import keras**

**from keras.layers import Dense**

**from keras.models import Sequential**

**from sklearn import metrics**

**import warnings**

**warnings.filterwarnings('ignore')**

**import numpy as np**

**import pandas as pd**

**import seaborn as sns**

**import matplotlib.pyplot as plt**

**import itertools as tt**

**dataset = pd.read\_csv('/Users/shaun/Downloads/kaggle/train.csv')**

**dataset**

**#Count of null values in a dataset**

**null\_columns=dataset.columns[dataset.isnull().any()]**

**combined\_data[null\_columns].isnull().sum()**

**dataset.mean()**

**kkk = dataset.drop('Cabin', axis=1)**

**kkk**

**#FILLING NAN WITH MEAN**

**cow = kkk.fillna(kkk.mean())**

**cow**

**#CREATING DUMMIES FOR CATERGORICAL DATA**

**sex = pd.get\_dummies(cow['Sex'],drop\_first=True)**

**embark = pd.get\_dummies(cow['Embarked'],drop\_first=True)**

**cow.drop(['Sex','Embarked','Name','Ticket'],axis=1,inplace=True)**

**cow = pd.concat([cow,sex,embark],axis=1)**

**cow.head()**

**#DROP NULL/MISSING VALUES**

**cow.dropna(inplace=True)**

**#CORRELATION SCATTER PLOT**

**sns.set(style="ticks")**

**sns.pairplot(cow,hue="Survived")**

**#CHECKING THE BALANCE OF THE DATA BY PLOTTING THE COUNT OUTCOMES BY THEIR VALUES**

**cow.dtypes.value\_counts()**

**sns.countplot(x='Survived', data=cow)**

**plt.show()**

**#A HEATMAP ILLUSTRATING THE RELATIONSHIP BETWEEN VARIABLE**

**sns.heatmap(cow[cow.columns[:12]].corr(),annot=True,cmap='bwr')**

**fig= plt.gcf()**

**fig.set\_size\_inches(10,8)**

**plt.show()**

**#CORRELATION BETWEEN VARIABLES**

**cow.corr()**

**#TRAIN TEST SPLIT**

**X\_train, X\_test, y\_train, y\_test = train\_test\_split(cow.drop(['Survived'],axis=1),**

**cow['Survived'], test\_size=0.25,**

**random\_state=101)**

**#TRAINING AND PREDICTING A DECISION TREE MODEL**

**dt\_model=DecisionTreeClassifier()**

**dt\_model.fit(X\_train,y\_train)**

**dt\_pred = dt\_model.predict(X\_test)**

**print(classification\_report(y\_test,dt\_pred))**

**#LOGISTIC REGRESSION MODEL**

**logmodel = LogisticRegression()**

**logmodel.fit(X\_train,y\_train)**

**predictions = logmodel.predict(X\_test)**

**X\_test.head()**

**#PREDICTION**

**Predictions**

**#CONFUSION MATRIX**

**print(confusion\_matrix(y\_test,predictions))**

**#ARTIFICIAL NUERAL NETWORK**

**ann = Sequential()**

**ann.add(Dense(units= 32,init= 'uniform', activation = 'relu', input\_dim=9))**

**ann.add(Dense(units= 32,init= 'uniform', activation = 'relu'))**

**ann.add(Dense(units= 1,init= 'uniform', activation = 'sigmoid'))**

**ann.compile(optimizer='adam',**

**loss='mean\_squared\_error',**

**metrics=['accuracy'])**

**ann.fit(X\_train,y\_train, batch\_size=32, nb\_epoch=300,verbose= 0)**

**#PREDICTION**

**ann\_pred = ann.predict(X\_test)**

**ann\_pred = [ 1 if y>=0.5 else 0 for y in ann\_pred]**

**print(ann\_pred)**

**#CONFUSION MATRIX**

**print(confusion\_matrix(y\_test,ann\_pred))**

**#CLASSIFICATION REPORT**

**print(classification\_report(y\_test,ann\_pred))**

**#IMPORTING THE TEST DATA TO TEST OUR PREDICTIONS**

**test = pd.read\_csv('/Users/shaun/Downloads/kaggle/test.csv')**

**test**

**test.drop('Cabin',axis=1,inplace=True)**

**test['Fare'].fillna(test['Fare'].median(), inplace=True)**

**test.head()**

**sex\_test = pd.get\_dummies(test['Sex'],drop\_first=True)**

**embark\_test= pd.get\_dummies(test['Embarked'],drop\_first=True)**

**test.drop(['Sex','Embarked','Name','Ticket'],axis=1,inplace=True)**

**test = pd.concat([test,sex\_test,embark\_test],axis=1)**

**test.head()**

**train.head()**

**ann.fit(train.drop(['Survived'],axis=1),train['Survived'] , nb\_epoch=300,verbose= 0)**

**test\_prediction = ann.predict(test)**

**test\_prediction = [ 1 if y>=0.5 else 0 for y in test\_prediction]**

**test\_pred = pd.DataFrame(test\_prediction, columns= ['Survived'])**

**new\_test = pd.concat([test, test\_pred], axis=1, join='inner')**

**new\_test.head()**

**final\_prediction = new\_test[['PassengerId' ,'Survived']]**

**final\_prediction.head()**