test1

May 12, 2021

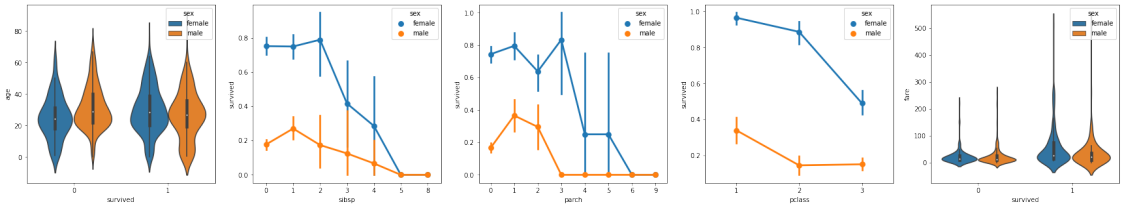
[4]:
```
import pandas as pd
import numpy as np
data = pd.read_csv('titanic.csv')
```

[5]:
```
data.replace('?', np.nan, inplace= True)
data = data.astype({"age": np.float64, "fare": np.float64})
#data.dtypes
```

[6]:
```
import seaborn as sns
import matplotlib.pyplot as plt

fig, axs = plt.subplots(ncols=5, figsize=(30,5))
sns.violinplot(x="survived", y="age", hue="sex", data=data, ax=axs[0])
sns.pointplot(x="sibsp", y="survived", hue="sex", data=data, ax=axs[1])
sns.pointplot(x="parch", y="survived", hue="sex", data=data, ax=axs[2])
sns.pointplot(x="pclass", y="survived", hue="sex", data=data, ax=axs[3])
sns.violinplot(x="survived", y="fare", hue="sex", data=data, ax=axs[4])
```

[6]:
```
<AxesSubplot:xlabel='survived', ylabel='fare'>
```



[7]:
```
data.replace({'male': 1, 'female': 0}, inplace=True)
```

[8]:
```
data.corr().abs()[["survived"]]
```

[8]:

	survived
pclass	0.312469
survived	1.000000
sex	0.528693
age	0.055513

```
sibsp     0.027825
parch     0.082660
fare      0.244265
```

[9]:
```
data['relatives'] = data.apply (lambda row: int((row['sibsp'] + row['parch']) >␣
↪0), axis=1)
data.corr().abs()[["survived"]]
```

[9]:
```
           survived
pclass     0.312469
survived   1.000000
sex        0.528693
age        0.055513
sibsp      0.027825
parch      0.082660
fare       0.244265
relatives  0.201719
```

[10]:
```
data = data[['sex', 'pclass','age','relatives','fare','survived']].dropna()
```

[11]:
```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test =␣
↪train_test_split(data[['sex','pclass','age','relatives','fare']], data.
↪survived, test_size=0.2, random_state=0)
```

[12]:
```
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X_train = sc.fit_transform(x_train)
X_test = sc.transform(x_test)
```

[13]:
```
from sklearn.naive_bayes import GaussianNB
model = GaussianNB()
model.fit(X_train, y_train)
```

[13]:
```
GaussianNB()
```

[14]:
```
### Test the model

from sklearn import metrics
predict_test = model.predict(X_test)
print(metrics.accuracy_score(y_test, predict_test))
```

```
0.7464114832535885
```

[15]:
```
# Alternatively use Neural Network

from keras.models import Sequential
```

```
from keras.layers import Dense

model = Sequential()
```

[16]:
```
model.add(Dense(5, kernel_initializer = 'uniform', activation = 'relu',␣
↪input_dim = 5))
model.add(Dense(5, kernel_initializer = 'uniform', activation = 'relu'))
model.add(Dense(1, kernel_initializer = 'uniform', activation = 'sigmoid'))
```

[17]:
```
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense (Dense)                (None, 5)                 30
_________________________________________________________________
dense_1 (Dense)              (None, 5)                 30
_________________________________________________________________
dense_2 (Dense)              (None, 1)                 6
=================================================================
Total params: 66
Trainable params: 66
Non-trainable params: 0
_________________________________________________________________
```

[18]:
```
model.compile(optimizer="adam", loss='binary_crossentropy',␣
↪metrics=['accuracy'])
model.fit(X_train, y_train, batch_size=32, epochs=50)
```

```
Epoch 1/50
27/27 [==============================] - 1s 22ms/step - loss: 0.6921 - accuracy:
0.5766
Epoch 2/50
27/27 [==============================] - 0s 14ms/step - loss: 0.6899 - accuracy:
0.5861
Epoch 3/50
27/27 [==============================] - 1s 25ms/step - loss: 0.6860 - accuracy:
0.5861
Epoch 4/50
27/27 [==============================] - 0s 13ms/step - loss: 0.6787 - accuracy:
0.5861
Epoch 5/50
27/27 [==============================] - 0s 14ms/step - loss: 0.6655 - accuracy:
0.5933
Epoch 6/50
27/27 [==============================] - 0s 13ms/step - loss: 0.6461 - accuracy:
```

```
0.6699
Epoch 7/50
27/27 [==============================] - 0s 14ms/step - loss: 0.6214 - accuracy:
0.7500
Epoch 8/50
27/27 [==============================] - 0s 17ms/step - loss: 0.5968 - accuracy:
0.7679
Epoch 9/50
27/27 [==============================] - 0s 16ms/step - loss: 0.5717 - accuracy:
0.7739
Epoch 10/50
27/27 [==============================] - 0s 17ms/step - loss: 0.5498 - accuracy:
0.7739
Epoch 11/50
27/27 [==============================] - 1s 42ms/step - loss: 0.5323 - accuracy:
0.7739
Epoch 12/50
27/27 [==============================] - 1s 20ms/step - loss: 0.5180 - accuracy:
0.7703
Epoch 13/50
27/27 [==============================] - 1s 24ms/step - loss: 0.5069 - accuracy:
0.7775
Epoch 14/50
27/27 [==============================] - 0s 15ms/step - loss: 0.4981 - accuracy:
0.7751
Epoch 15/50
27/27 [==============================] - 1s 19ms/step - loss: 0.4915 - accuracy:
0.7799
Epoch 16/50
27/27 [==============================] - 0s 18ms/step - loss: 0.4870 - accuracy:
0.7739
Epoch 17/50
27/27 [==============================] - 1s 36ms/step - loss: 0.4818 - accuracy:
0.7787
Epoch 18/50
27/27 [==============================] - 1s 28ms/step - loss: 0.4779 - accuracy:
0.7775
Epoch 19/50
27/27 [==============================] - 0s 14ms/step - loss: 0.4744 - accuracy:
0.7787
Epoch 20/50
27/27 [==============================] - 1s 32ms/step - loss: 0.4714 - accuracy:
0.7823
Epoch 21/50
27/27 [==============================] - 0s 18ms/step - loss: 0.4690 - accuracy:
0.7811
Epoch 22/50
27/27 [==============================] - 0s 11ms/step - loss: 0.4668 - accuracy:
```

```
0.7811
Epoch 23/50
27/27 [==============================] - 1s 19ms/step - loss: 0.4645 - accuracy:
0.7859
Epoch 24/50
27/27 [==============================] - 0s 15ms/step - loss: 0.4628 - accuracy:
0.7811
Epoch 25/50
27/27 [==============================] - 0s 11ms/step - loss: 0.4612 - accuracy:
0.7799
Epoch 26/50
27/27 [==============================] - 1s 23ms/step - loss: 0.4603 - accuracy:
0.7811
Epoch 27/50
27/27 [==============================] - 1s 35ms/step - loss: 0.4588 - accuracy:
0.7835
Epoch 28/50
27/27 [==============================] - 1s 23ms/step - loss: 0.4574 - accuracy:
0.7883
Epoch 29/50
27/27 [==============================] - 1s 19ms/step - loss: 0.4560 - accuracy:
0.7859
Epoch 30/50
27/27 [==============================] - 1s 39ms/step - loss: 0.4550 - accuracy:
0.7847
Epoch 31/50
27/27 [==============================] - 0s 11ms/step - loss: 0.4542 - accuracy:
0.7835
Epoch 32/50
27/27 [==============================] - 1s 20ms/step - loss: 0.4535 - accuracy:
0.7835
Epoch 33/50
27/27 [==============================] - 0s 14ms/step - loss: 0.4524 - accuracy:
0.7835
Epoch 34/50
27/27 [==============================] - 0s 11ms/step - loss: 0.4517 - accuracy:
0.7883
Epoch 35/50
27/27 [==============================] - 2s 67ms/step - loss: 0.4522 - accuracy:
0.7895
Epoch 36/50
27/27 [==============================] - 0s 11ms/step - loss: 0.4502 - accuracy:
0.7859
Epoch 37/50
27/27 [==============================] - 1s 21ms/step - loss: 0.4504 - accuracy:
0.7847
Epoch 38/50
27/27 [==============================] - 0s 14ms/step - loss: 0.4494 - accuracy:
```

```
0.7835
Epoch 39/50
27/27 [==============================] - 0s 17ms/step - loss: 0.4490 - accuracy:
0.7859
Epoch 40/50
27/27 [==============================] - 0s 17ms/step - loss: 0.4488 - accuracy:
0.7835
Epoch 41/50
27/27 [==============================] - 0s 17ms/step - loss: 0.4484 - accuracy:
0.7859
Epoch 42/50
27/27 [==============================] - 0s 13ms/step - loss: 0.4487 - accuracy:
0.7811
Epoch 43/50
27/27 [==============================] - 0s 17ms/step - loss: 0.4479 - accuracy:
0.7823
Epoch 44/50
27/27 [==============================] - 1s 33ms/step - loss: 0.4477 - accuracy:
0.7859
Epoch 45/50
27/27 [==============================] - 1s 22ms/step - loss: 0.4469 - accuracy:
0.7883
Epoch 46/50
27/27 [==============================] - 1s 52ms/step - loss: 0.4471 - accuracy:
0.7895
Epoch 47/50
27/27 [==============================] - 1s 24ms/step - loss: 0.4466 - accuracy:
0.7883
Epoch 48/50
27/27 [==============================] - 2s 72ms/step - loss: 0.4465 - accuracy:
0.7871
Epoch 49/50
27/27 [==============================] - 1s 34ms/step - loss: 0.4461 - accuracy:
0.7883
Epoch 50/50
27/27 [==============================] - 0s 17ms/step - loss: 0.4464 - accuracy:
0.7895
```

[18]:
```
<tensorflow.python.keras.callbacks.History at 0x7fc7903b77f0>
```

[19]:
```
y_pred = model.predict_classes(X_test)
print(metrics.accuracy_score(y_test, y_pred))
```

```
WARNING:tensorflow:From <ipython-input-19-6007bb8697bc>:1:
Sequential.predict_classes (from tensorflow.python.keras.engine.sequential) is
deprecated and will be removed after 2021-01-01.
Instructions for updating:
Please use instead:* `np.argmax(model.predict(x), axis=-1)`,   if your model
```

does multi-class classification (e.g. if it uses a `softmax` last-layer activation).* `(model.predict(x) > 0.5).astype("int32")`, if your model does binary classification (e.g. if it uses a `sigmoid` last-layer activation).
0.8038277511961722

[ ]: