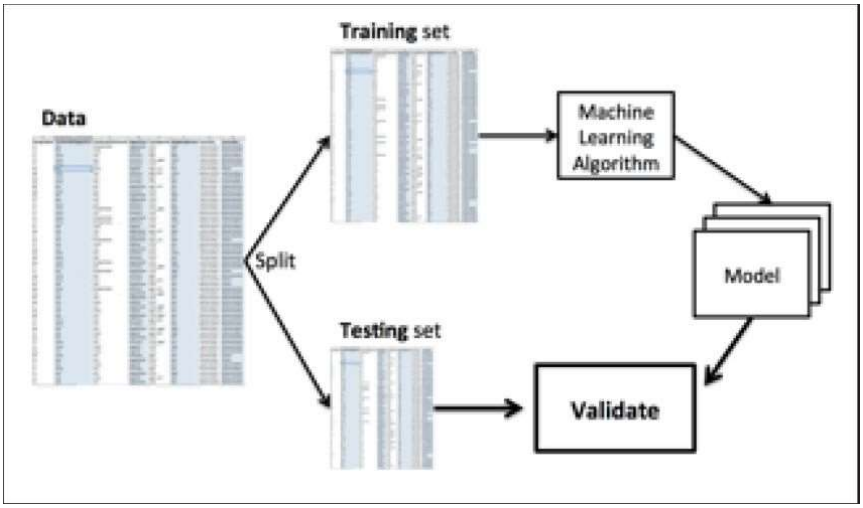
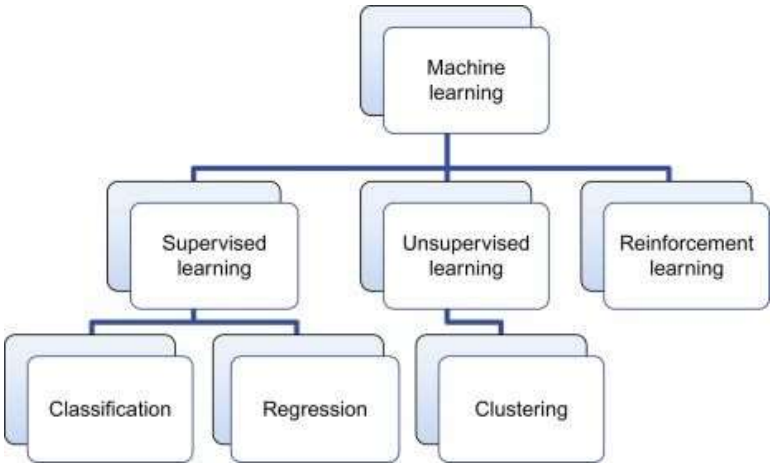
Systems with Machine Learning
Task 4: Testing
Topic: Wine Quality Analysis

1. **Knowing which machine learning methods you use, describe possible testing metrics and procedures (accuracy, precision, recall, sensitivity, F-score, etc.)**





For the given problem we identified two methods for solving the problem, regression or classification. Both methods fall within the category of supervised ML. Regression helps to predict or explain a particular numerical value based on a set of our prior data, Classification methods predict or explain a class value. Regression algorithms attempt to estimate the mapping function from the input variables to numerical or continuous output variables. On the other hand, Classification attempts to estimate the mapping function from the input variables to discrete or categorical output variables.

Since our data our dataset consists of integer quality grades for each wine so we decided to solve this problem as a classification problem. Each grade from 0-10 corresponds to one glass of wine. In such a way we mitigate the problem of getting a result of wine quality that is a floating-point number. E.g. Quality = 5.1423

For calculating a loss value we are going to use a categorical cross-entropy function. Cross-entropy loss, or log loss, measures the performance of a classification model whose output is a probability value between 0 and 1. Cross-entropy loss increases as the predicted probability diverges from the actual label. So predicting a probability of .012 when the actual observation label is 1 would be bad and result in a high loss value. A perfect model would have a log loss of 0.

In binary classification, where the number of classes M equals 2, cross-entropy can be calculated as:

$$-(\log() + (1-)\log(1-))$$

If $M>2$ (i.e. multiclass classification), we calculate a separate loss for each class label per observation and sum the result.

$$-\sum_{c=1}^{M} y_{o,c} \log(p_{o,c})$$

M − number of classes (dog, cat, fish)
Log − the natural log
Y − binary indicator (0 or 1) if class label is the correct classification for observation
p − predicted probability observation is of class

2. **Choose few (more than one) methods/metrics which are suitable to your problem. Justify why.**

After the training and briefly examining the results of the TEST subset we can come to the conclusion that the wine quality classification may not necessarily be a problem for neural networks. We observed in the training of the SPLIT2 the network fits very well with the training data but it struggled to get a proper result for anything beyond the TRAIN subset. It may be caused by the subjective nature of the wine qualifying. The dataset consists of around 6500 ratings, which simply could not have been assessed by one person. Maybe in the dataset, there are some conflicting records that give totally different ratings for similar wines.

Nevertheless, we promised that some further analysis of the network efficiency will be conducted as a task 4 of this project. From a technical point of view, we found out that we could combine red and white wines together by adding a binary attribute that would indicate a color, e.g. 0 - white, 1 - red. Ultimately, we decided to keep them separate as the network had a problem with classifying a single color of the wine.

```
### PARAMETERS TO BE EXPERIMENTED WITH ###
LAYER_UNITS_1 = 120 #number of neurons in the first layer of neural network
LAYER_UNITS_2 = 80  #number of neurons in the second layer of neural network
EPOCHS = 300        #number of iterations after which the learning finishes
BATCH_SIZE = 32     #batch size after which weights are updated
```

```
1119/1119 [==============================] - 0s 27us/step
TRAIN DATASET:              loss 0.032      accuracy 0.994
240/240 [==============================] - 0s 28us/step
VALIDATION DATASET:         loss 1.985      accuracy 0.663
240/240 [==============================] - 0s 51us/step
TEST DATASET:               loss 2.620      accuracy 0.688
```

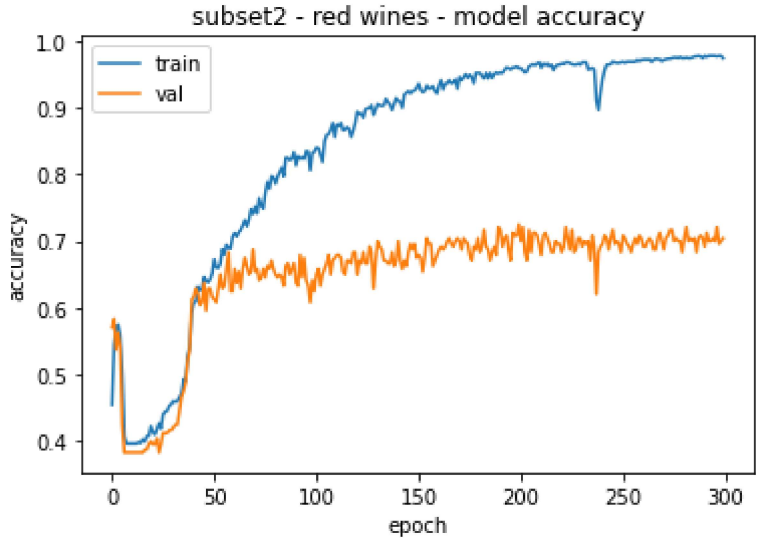
The above report suggests that network accuracy of the model is not much satisfactory as it has not been able to achieve anything better with TEST and VALIDATION subset. Our bid therefore is to improve the neural network by modifying parameters like batch_size, layer_size etc and hoping for a satisfactory result accuracy.

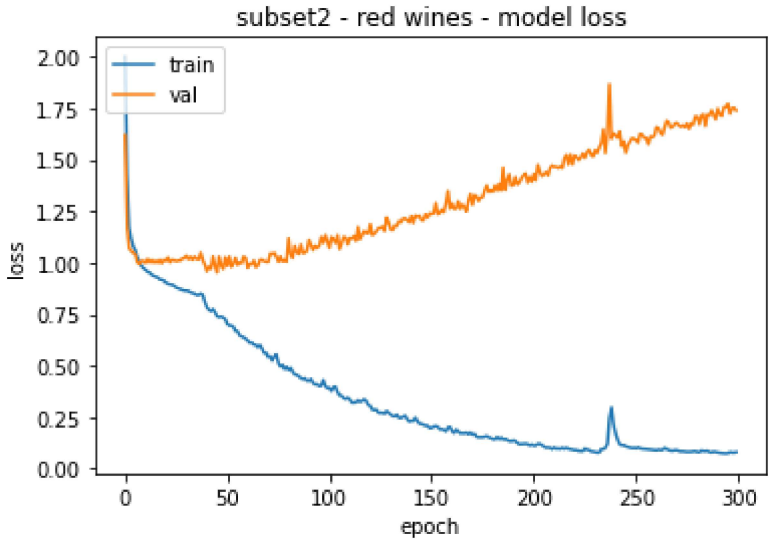
LAYER_SIZE MODIFICATION

When we performed modifications with the layer_unit 1 and layer_unit 2 = (200,200), max_iter=300. We found Accuracy is `0.701` and keeping iteration constant for all the sizes. It found it's better than the previous result.

When I changed the layer_units 2 = (200,300), I found interestingly that the Accuracy is 0.721.

```
1119/1119 [==============================] - 0s 34us/step
TRAIN DATASET:    loss 0.076  accuracy 0.978
240/240 [==============================] - 0s 43us/step
VALIDATION DATASET:    loss 1.741  accuracy 0.704
240/240 [==============================] - 0s 36us/step
TEST DATASET:    loss 1.970  accuracy 0.721
```





We feel that we are not satisfied with this accuracy. What else can we do apart from "Neural Networks"?

Let's try with "MinMaxScalar" using "scikit-learn"

```
from sklearn.preprocessing import MinMaxScaler
MinMaxScaler(feature_range=(0, 1), copy=True)
minmax=MinMaxScaler(feature_range=(0, 1), copy=True)

redwine=pd.read_csv("wine-quality-red.csv", sep=",")
redwine

## to transform full data to minmax
scale=minmax.fit_transform(redwine)

x = redwine.iloc[:,[0,1,2,3,4,5,6,7,8,9,10]].values
y = redwine.iloc[:,11].values
y=np.reshape(y,newshape = (-1,1))
x=minmax.fit_transform(x)
print(x)

np.asarray(x, dtype=float)

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=0)

scalar = MLPClassifier(layer_sizes=(200,300), max_iter=300)
print(scalar.fit(x,y))
```

```
scalar_pred_class=scalar.predict(X_test)
print(scalar_pred_class)

ac_scale = accuracy_score(Y_test,scalar_pred_class)

print(ac_scale)
```

Combine Results for Testing

Accuracy after usage of MINMAX Scale has increased from 0.721 to 0.822 for the same neural network classifier having the two layer_sizes = (200,300), max_iter=300. Now we can say our prediction model is good.

However, though we got the highest accuracy score with the MInMax Scaling, it is still not an ideal model because the results are not representative enough for all of the areas. In this case, and in order to improve the model, we need to try other metrics.