Lab 3: Grid search and Clustering

In this lab you will continue to perform basic machine learning tasks. In the previous Lab you performed a linear regression from scratch. In this Lab you will perform a linear regression using the scikit-learn library and contrast the results with the ones obtained in Lab 2. You will also perform polynomial regression with Scikit-learn and hyper-parameter optimization to find the optimal degree of the polynomial. Then you will perform a K-means classification analysis on a new dataset from scratch, then you will use grid search to find the optimal number of clusters. Finally, you will predict the belonging cluster for 3 new datapoints.

Load once again the datasets you used to create the linear regression and the new **Average rent in a rental apartment by year and region.** All these datasets can be found in the canvas assignment page.

Task 1: Linear regression with scikit-Learn

Load the datasets from the previous lab. This time you are going to perform the linear regression using the library scikit-learn.

- Obtain the regression coefficients.
- Once again predict for the following values (35, 80)
- Finally obtain the MSE value
- Compare the results with the ones obtained in the previous lab.
- Are there any differences?

Task 2: Polynomial regression with hyperparameter optimization

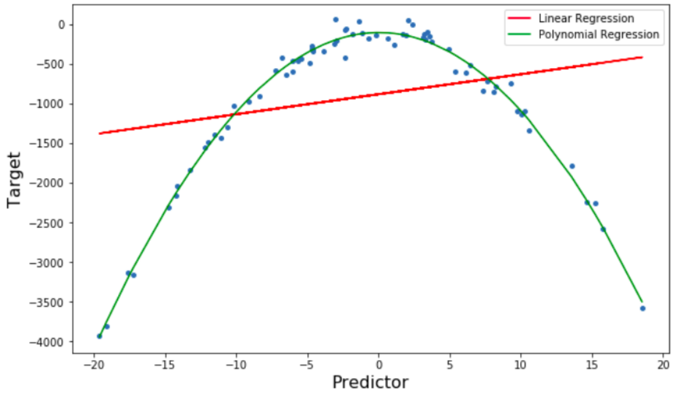
In both Task 1 and Lab 2 you performed linear regression to find the relationship between age and income. However, the linear regression result may not be the optimal result as shown by the MSE value.

You will now use Scikit-learn to perform polynomial regression and evaluate if it is a better model.

- 2.1: Polynomial regression is a special case of linear regression. Perform polynomial regression. With the use of **PolynomialFeatures** from **sklearn.preprocessing** you will be able to increase the number of features the linear regression model is trained.
    - For additional information on how to perform this task check lecture 8 and page 128 of the ML book.
- 2.2: You may notice there is a Hyperparameter for this model, which is the polynomial degree, you will not need to find the optimal degree of this polynomial using Gridsearch.
    - For this task you will need to implement Gridsearch from scratch, you will be able to use scikit-learn for fitting the data, but you will manually iterate over a set of degrees and find the cross-validation score for each degree to find the optimal degree value.
    - Which order of the polynomial is best?

- 2.3: Graph the results of the polynomial regression line with the optimal degree found along with the linear regression line.

An example of the desired graph that ***is not done on your dataset*** but shows the type of graph you need to present is the following:



Compare the results from Task 2 and Task 1, which model is the best and why?

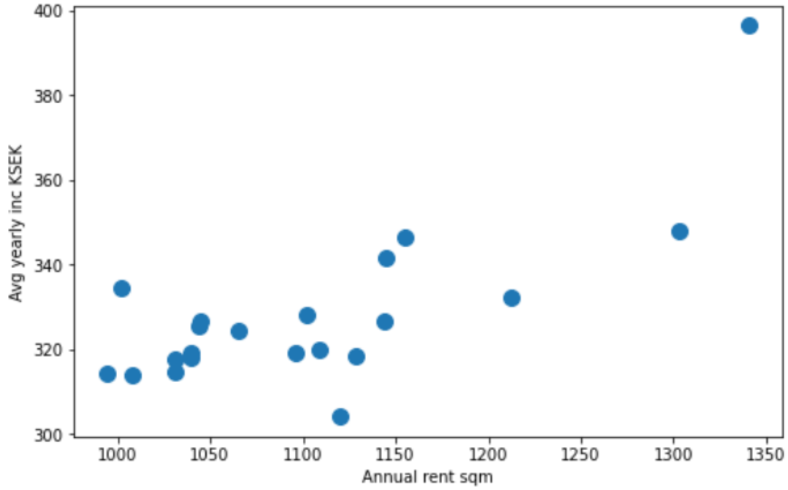
Task 3: Clustering with K-means

- 3.1: Load the new dataset found in the canvas page for this Lab: **rent_vs_inc.csv** the same way you did in the last lab. The first few rows of this dataset should look like this:

```
   year                   region  Annual rent sqm  Avg yearly inc KSEK
0  2020        01 Stockholm county             1341           396.428571
1  2020          03 Uppsala county             1303           347.985714
2  2020     04 Södermanland county             1129           318.292857
3  2020     05 Östergötland county             1144           326.757143
4  2020        06 Jönköping county             1044           325.521429
```

What information can you get by just looking at the table?

- 3.2: Start by creating a scatter plot for your points.
- Scatter plot example:



- 3.3: As mentioned in the introduction section you will create your K-means function from scratch. For information on how to perform K-means can be found in lecture 11:
    - Initialize the centroids with a starting number of clusters you consider correct, the centroids can be selected as a random point among your sample.
    - Find which point belongs to which cluster by finding the closer centroid to every point (Euclidean distance).
    - Calculate the mean point among each cluster to obtain the new centroid
    - Repeat this process a N (around 10) number of iterations until the mean of the new cluster does not change from the previous iteration.
    - Create once again a cluster plot with colors assigned for each cluster. For information on how to do this plot, check: https://pythonguides.com/matplotlib-scatter-plot-color/

Task 4: Grid search for hyper-parameter optimization

In this task you will implement hyper-parameter optimization to find the optimal number of k-means clusters for this classification task.

- 4.1: In order to perform the hyper parameter optimization for the number of clusters you will use the silhouette score, for this score create a function that finds the silhouette coefficient for each of the clusters and plot them.
    - For more information on the silhouette score and how it can be obtained with Scikit-learn, check page 247 of the ML book.
    - The silhouette coefficient for each point (i) can be obtained as follows:
    - $$S(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}}$$
    - Where:
        - S(i) is the silhouette coefficient of the data point i.
        - a(i) is the average distance between i and all the other data points in the cluster to which i belong.
        - b(i) is the average distance from i to all the points to the closest cluster to which i does not belong.
    - Create a function a(i) that calculates the average intra-cluster distance from any point i.
    - Create a function that calculates the average inter-cluster distance to the closets cluster from datapoint i.
    - Create a main function that iterates over all points in the dataset and calculates S(i)
    - Finally obtain the average S(i) for all points for each cluster value to get the cluster's silhouette coefficient.
    - Graph the cluster's silhouette coefficient such as the example in the book does to find the optimal number of clusters.
- 4.2: Create a scatter plot with the cluster colors for the newly found number of clusters.
- 4.3: Finally evaluate how your created model predicts new values:

- For this task we will assume we have 3 unnamed regions with the following annual rent and average salary [**`1010, 320.12], [1258, 320], [980, 292.4]`**
- Find which cluster these data points belong to and plot them in the graph.
- Based on the cluster graph do you think your model successfully predicted the cluster for these values?

Optional advanced task

Can you figure out how to make an N-dimensional grid search optimizer, which can handle an arbitrary number of hyperparameters to optimize?