**Practical – Cross Validation, Feature Selection and Ford Cars**

In this practice, we will learn how to train with **CV (Cross-Validation)** and how to choose features with **Feature Selection** algorithms.  
We will use [**sweetviz**](https://pypi.org/project/sweetviz/) to show the DataFrame report.  
We will also use [**tqdm**](https://tqdm.github.io/) to show a progress bar.

**Downloads, Imports, and Definitions**

As usual, update packages whose Colab version is old, plotly, pandas\_profiling  ,sweetviz.

And import all our former practicals’ packages.

**Data Exploration**

In this practical we use a dataset of used Ford cars.

**Dataset Information**

The types of the cars are:

**SE**   
*2020 Ford Fusion Hybrid SE FWD*

**SES**   
*2020 Ford EcoSport SES 4WD*



**SEL**  
*2020 Ford Edge SEL FWD*

**Attribute Information**

1. **year**: the year of manufacturing (2000-2011)
2. **model**: the model of the car (SE, SES, SEL)
3. **price**: the price of the car (3800-21992)
4. **mileage**: the number of mileage that the car has done (4867-151479)
5. **color**: the color of the car ('Yellow', 'Gray', 'Silver', 'White', 'Blue', 'Black', 'Green', 'Red', 'Gold')
6. **transmission**: the transmission of the car ('AUTO', 'MANUAL')

**Target Information**

* **price**: The target is the price! (a regression problem)



Download the dataset from [Github](https://raw.githubusercontent.com/stedy/Machine-Learning-with-R-datasets/master/usedcars.csv) and explore it with Pandas tools – print its description and info.

Another way to show a useful report on the data is sweetviz analysis. This report is lighter than pandas\_profiling so it may work better on large datasets.

Import sweetviz and print its report on the usedcars df

Sweetviz can also produce a *comparative* report of two subsets of the data.

Compare the AUTO transmission and the MANUAL transmission subsets.

**Cross-Validation**

Cross-Validation (CV) is used instead of splitting the training data just to train and validation subsets.   
This method provides better predictions of the test set results.  
The technique is similar to out former data splitting, but in this case we need to ensure that the split does not affect our result , by trying out several possible splits.

The two CV methods you’ll use here are:

1. **KFold**
2. **LPO** (Leave P out)

**KFold** is a faster technique.  
**LPO** is more precise and provides better predictions of the test score.

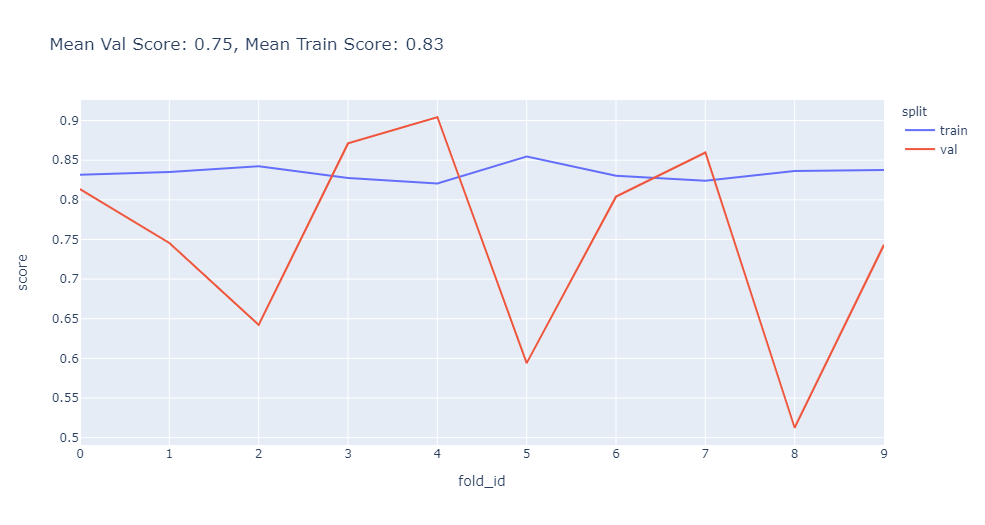
Use Scikit-learn [KFold](https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.KFold.html#sklearn.model_selection.KFold) to split the data into k=5 folds.(remember to separate the data to features and target). Print the 5 folds.

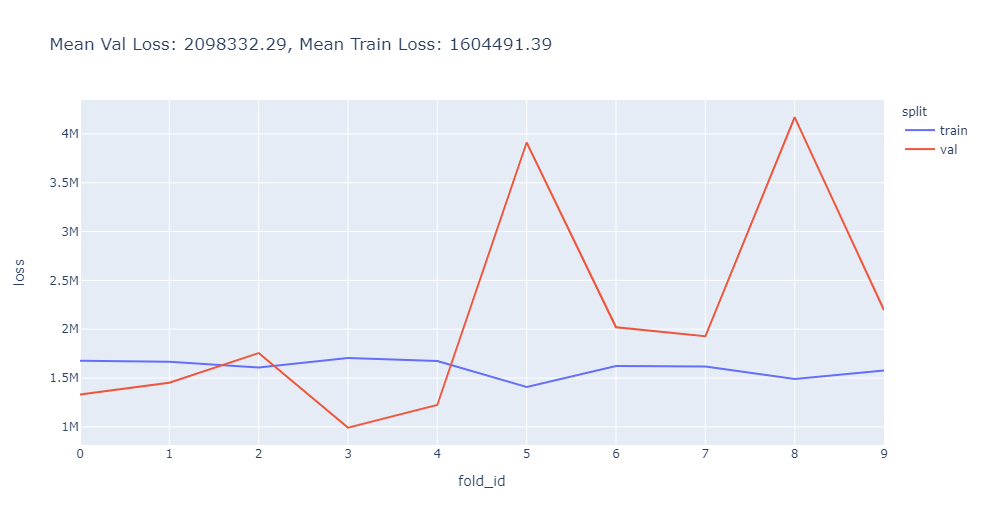
Now use Scikit-learn [LeavePOut](https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.LeavePOut.html), with leaving out  p=3.   
Note: LPO can be used as LOO (Leave **One** Out) just by specifying p=1.

Use [tqdm](https://tqdm.github.io/) to show a progress bar.

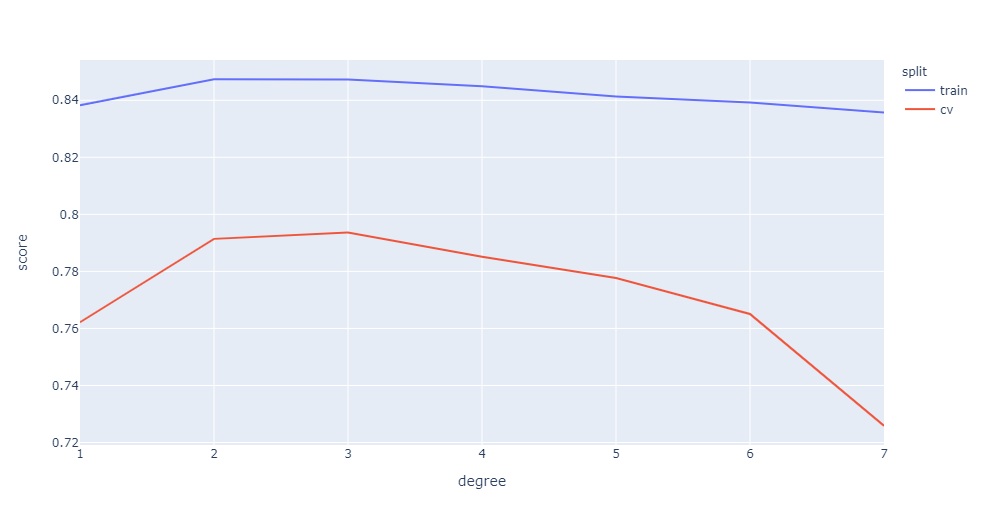
Use previous practicals’ codes and create a method that inputs dataset and model and performs the regression and returns R2 score and MSE loss.

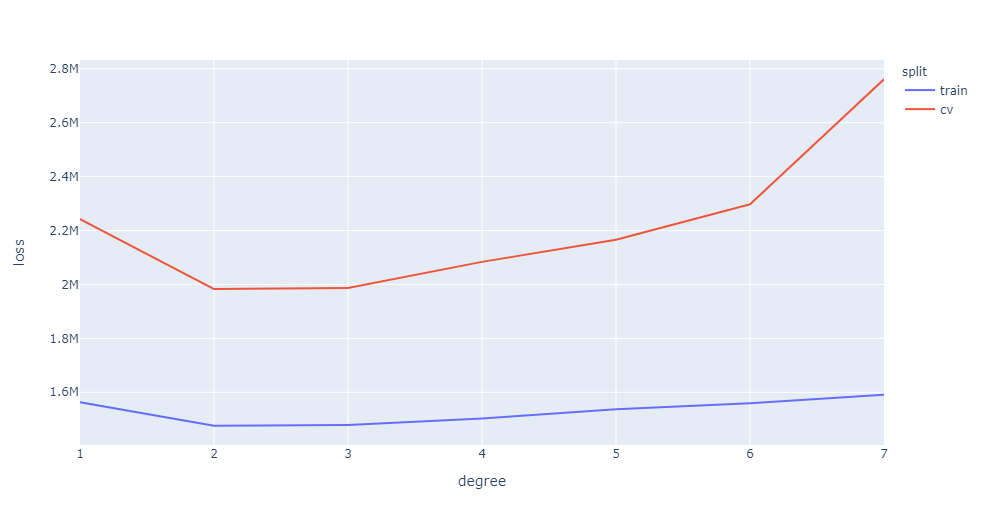
Repeat it 10 times, each time splitting the data again for the CV. Put an option in the code to use either K-fold or LPO. Print and plot graph of the train and validation scores and loss as a function of repeated trial. Print the mean scores over the CV 10 trials for both the train and the validation sets.





Use this function to calculate, print and plot graph of the train and validation scores as a function of different polynomial degrees (try degrees 1 to 7). Do it for the polynomial numerical features only (remember - we looked at feature types.. ).





What is your interpretation of the graph ? which is the best degree?

**Feature Selection (we haven’t studied it in class, but may be useful for your assignment..)**

To reduce dimensionality and increase accuracy, it is good to choose the best features for our use-case.  
There are 3 methods of Feature Selection:

1. **Forward Feature Selection**
2. **Backward Feature Selection**
3. **Hybrid Feature Selection**

**Forward Feature Selection** starts from zero features and adds features until it reaches either a maximal features or the best score.

**Backward Feature Selection** starts from the full feature set and removes features until it reaches either a maximal features or the best score

**Hybrid Feature Selection**, starts from zero features and add or remove features until it reaches the best score.

The Scikit-learn [RFE](https://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.RFE.html#sklearn.feature_selection.RFE) is based on the **Backward Feature Selection**.  
Specify the target number of features and the selector will stop when it reaches this number of features.

Find the best n\_features\_to\_select=3 features in the cars dataset, for SGDRegressor.

In [17]:

*# choose the best 3 features of this dataset with SGDRegressor*

**from** **sklearn.feature\_selection** **import** RFE

numerical\_cols = X.select\_dtypes(include=['int64', 'float64']).columns

categorical\_cols = X.select\_dtypes(include=['object', 'bool']).columns

all\_cols = categorical\_cols.tolist() + numerical\_cols.tolist()

ct\_enc\_std = ColumnTransformer([

("encoding", OrdinalEncoder(), categorical\_cols),

("standard", StandardScaler(), numerical\_cols)])

X\_encoded = pd.DataFrame(ct\_enc\_std.fit\_transform(X, t), columns=all\_cols)

selector = RFE(SGDRegressor(random\_state=1), n\_features\_to\_select=3).fit(X\_encoded, t)

X\_encoded.loc[:, selector.support\_]

Out[17]:

|  | **model** | **year** | **mileage** |
| --- | --- | --- | --- |
| **0** | 1.0 | 1.036340 | -1.370208 |
| **1** | 1.0 | 1.036340 | -1.239574 |
| **2** | 1.0 | 1.036340 | -1.372513 |
| **3** | 1.0 | 1.036340 | -1.214028 |
| **4** | 0.0 | 1.492208 | -1.334733 |
| **...** | ... | ... | ... |
| **145** | 2.0 | -1.243000 | 1.886781 |
| **146** | 0.0 | -3.066473 | 1.589407 |
| **147** | 0.0 | -3.978209 | 1.955240 |
| **148** | 0.0 | -3.522341 | 3.986996 |
| **149** | 0.0 | -3.978209 | 2.417013 |

150 rows × 3 columns

Which are the best 3 features you got?

->We can see that the best 3 features are model, year, and mileage.

[RFECV](https://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.RFECV.html) in Scikit-learn uses CV **and** chooses the best number of features on this dataset.  
The default CV is 5-fold cross-validation.

Use it, again with SGDRegressor and enter the Scikit-learn [RepeatedKFold](https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.RepeatedKFold.html) to repeat each KFold 10 times with different splits. Plot a graph of the cross validation score for each number of selected features, from 1 to 5.

In [18]:

*# find best subset of features on this dataset*

**from** **sklearn.feature\_selection** **import** RFECV

**from** **sklearn.model\_selection** **import** RepeatedKFold

numerical\_cols = X.select\_dtypes(include=['int64', 'float64']).columns

categorical\_cols = X.select\_dtypes(include=['object', 'bool']).columns

all\_cols = categorical\_cols.tolist() + numerical\_cols.tolist()

ct\_enc\_std = ColumnTransformer([

("encoding", OrdinalEncoder(), categorical\_cols),

("standard", StandardScaler(), numerical\_cols)])

X\_encoded = pd.DataFrame(ct\_enc\_std.fit\_transform(X, t), columns=all\_cols)

selector = RFECV(SGDRegressor(random\_state=1), cv=RepeatedKFold(n\_splits=5, n\_repeats=10, random\_state=1)).fit(X\_encoded, t)

display(X\_encoded.loc[:, selector.support\_])

fig = go.Figure()

fig.add\_trace(go.Scatter(x=[i **for** i **in** range(1, len(selector.grid\_scores\_) + 1)], y=selector.grid\_scores\_))

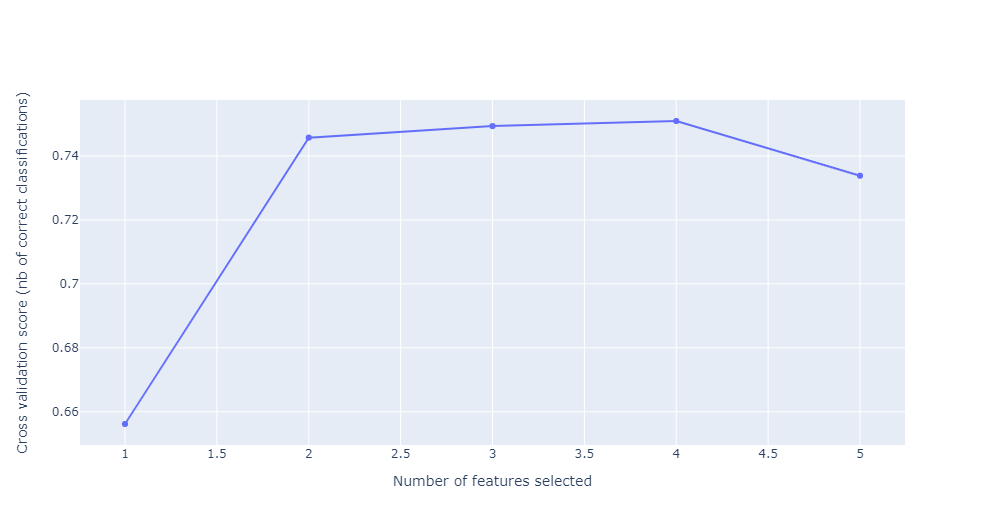
fig.update\_xaxes(title\_text="Number of features selected")

fig.update\_yaxes(title\_text="Cross validation score (nb of correct classifications)")

fig.show()

|  | **model** | **color** | **year** | **mileage** |
| --- | --- | --- | --- | --- |
| **0** | 1.0 | 8.0 | 1.036340 | -1.370208 |
| **1** | 1.0 | 3.0 | 1.036340 | -1.239574 |
| **2** | 1.0 | 6.0 | 1.036340 | -1.372513 |
| **3** | 1.0 | 3.0 | 1.036340 | -1.214028 |
| **4** | 0.0 | 7.0 | 1.492208 | -1.334733 |
| **...** | ... | ... | ... | ... |
| **145** | 2.0 | 6.0 | -1.243000 | 1.886781 |
| **146** | 0.0 | 5.0 | -3.066473 | 1.589407 |
| **147** | 0.0 | 5.0 | -3.978209 | 1.955240 |
| **148** | 0.0 | 8.0 | -3.522341 | 3.986996 |
| **149** | 0.0 | 5.0 | -3.978209 | 2.417013 |

150 rows × 4 columns



The best subset of features is model, color, year, and mileage.  
It is possible to use the best model, already fitted to the data, with the selector.estimator\_ attribute.

**More Information**

Explanation on how to use KFold and LPO (Leave P Out):  
[Cross-validation: evaluating estimator performance](https://scikit-learn.org/stable/modules/cross_validation.html)

An answer on how to choose k in KFold:  
[Choice of K in K-fold cross-validation](https://stats.stackexchange.com/questions/27730/choice-of-k-in-k-fold-cross-validation)

An answer to the differences between KFold and LOO (Leave One Out):  
[10-fold Cross-validation vs leave-one-out cross-validation](https://stats.stackexchange.com/questions/154830/10-fold-cross-validation-vs-leave-one-out-cross-validation)

An answer on how to change tqdm bar size:  
[How to change tqdm's bar size](https://stackoverflow.com/questions/54362541/how-to-change-tqdms-bar-size)

An advanced example of Scikit-learn transformers:  
[Can You Consistently Keep Track of Column Labels Using Sklearn's Transformer API?](https://stackoverflow.com/a/57534118)

A Guide on how to use all CV algorithms of Sckikit-learn:  
[Cross-validation: evaluating estimator performance](https://scikit-learn.org/stable/modules/cross_validation.html)

Scikit-learn removed Bootstrap from the CV options:  
[What should I use instead of Bootstrap?](https://stackoverflow.com/questions/28030291/what-should-i-use-instead-of-bootstrap)

A function transformer in Scikit-learn:  
[sklearn.preprocessing.FunctionTransformer](https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.FunctionTransformer.html)

Documentation on Scikit-learn Feature Selection:  
[Feature selection](https://scikit-learn.org/stable/modules/feature_selection.html)