A logo for college computing

Description automatically generated

**Assessment Cover Page**

|  |  |
| --- | --- |
| *Student Full Name: Cristhian Elson Pereira Macedo* |  |
| *Student Number: 2024104* |  |
| *Module Title: Machine Learning (10 ETCS)* |  |
| *Assessment Title: CA1 Project* |  |
| *Assessment Due Date: 21st April 2024 23:59* |  |
| *Date of Submission: 21st April 2024 22:57* |  |

**Declaration**

By submitting this assessment, I confirm that I have read the CCT policy on academic misconduct and understand the implications of submitting work that is not my own or does not appropriately reference material taken from a third party or other source.

I declare it to be my own work and that all material from third parties has been appropriately referenced.

I further confirm that this work has not previously been submitted for assessment by myself or someone else in CCT College Dublin or any other higher education institution.



**House Predict Prices**

**Cristhian Elson Pereira Macedo**

**2024104**

Higher Diploma in Science in Data Analytics for Business

Machine Learning (MLDA)

Lecturer: Dr. Muhammad Iqbal

CCT College Dublin

Dublin, Ireland

2024



**SUMMARY**

[**1.** **Introduction** 4](#_Toc164632448)

[**2.** **Data Understanding** 5](#_Toc164632449)

[**3.** **Data Preparation** 5](#_Toc164632450)

[**4.** **Machine Learning tecniquies** 7](#_Toc164632451)

[**a)** **KNN Classification** 7](#_Toc164632452)

[**b)** **Random Forest** 8](#_Toc164632453)

[**c)** **Decision Trees** 8](#_Toc164632454)

[**d)** **Linear Regression** 8](#_Toc164632455)

[**i.** **Linear Regression Hypothesis** 9](#_Toc164632456)

[**e)** **K-Means** 9](#_Toc164632457)

[**f)** **Cross-Validation and GridSearchCV** 9](#_Toc164632458)

[**5.** **Conclusion** 10](#_Toc164632459)

[**References** 12](#_Toc164632460)

# **Introduction**

This project aims to complete the first Assessment Task for the module Machine Learning from the course Higher Diploma in Science in Data Analytics for Business by CCT College Dublin, this is a document to describe the project itself whose name is “CristhianMacedo\_MLearningHDip\_CA1.ipynb” and should be always with this document. This project for Machine Learning uses the programming language Python, the environment of Anaconda Navigator with Jupyter Notebook, with CRISP-DM methodology as project management.

Over the years everything started to change some things for good and others not as much, and this situation is one case, about house prices, as a resident of Ireland who pays rent, and dreams of buying a house in the future, it is an interesting way of learning and predicting some house prices around the region.

Therefore the area chosen to be covered here is Housing and Zoning and it will address an Exploratory Data Analysis, with some statistics analyses focused on developing and deploying Machine Learning to predict prices.

Once the data is small and contains a few features there is no need to use feature reduction algorithms such as Principal Component Analysis (PCA), but it will be analysed using KNN Classification, Random Forest, Decision Trees, Linear Regression, K-Means and some analyses using Cross-Validation and GridSearchCV to find optimal parameters.

After some research on the website Central Statistics Office, Ireland (cso.ie) it was found interesting data about Residential Property until January 2024, “HPM04 Market-based Household Purchases of Residential Dwellings” (Central Statistics Office, 2024).

This database contains some data about “Market-based Household Purchases of Residential Dwellings by Dwelling Status, Eircode Output, Stamp Duty Event, Type of Buyer, Month (2010 January - 2024 January)” (Central Statistics Office, 2024), and is updating month by month. For this project, it was chosen the data about Mean Sale Price in Euro, All dwelling statuses, stamp duty events: Executions, months from 2010 January to 2024 January, all Eircode output from the counties of Ireland and type of buyer just the First-Time buyer owner-occupier.

# **Data Understanding**

After getting and loading the database using the Pandas library using “.head()” it was possible to get the first impression of the data, returning that this dataset has 8 columns (features) and after using “.shape” returning the shape of it, in this case, 23491 rows (observations) to be analysed.

Also used “.info()” to get a summary of the Data Frame, and next “.describe()” to see statistical information of the numerical features, in this case, it appears only once, for a small data frame. Checked the types of data for each feature using “.dtypes” and “.unique()” for every feature, to understand them better and until this point, looks like ok.

# **Data Preparation**

In this stage, it was checked some information about the features using “.columns” and “.value\_counts()”, next checking if any missing values existed in the features with the command “.isnull().sum()” and got the result that there are 3189 observations in this feature missing values that are approximately 14% of 100% values in this feature, also used a bar graphic with missingno library to visualise it.

Not find any syntax errors or divergence between the features, but found an interesting approach to apply and split the feature Eircode output into two parts, separating Acronym county’s name and County in new features.

Some features it was considered irrelevant, so from 10 features existing until this moment, opted to drop 5 of them, being: Statistic Label, Dwelling Status, Stamp Duty Event, Type of Buyer and UNIT, and renamed the existent features Eircode Output to Eircode and VALUE to Value.

Time to handle missing values, and first using “.replade()” to patronise all different characters in the data as NaN (null) to be easy to handle next. But before imputing these missing values opted to create hypotheses instead and create a new feature whose name will be Sold.

Assuming that there are no values in these observations because the properties were sold and after sold they can no longer present values as they are private data.

Opted to do this, once the data frame is there are not many features to be analysed and used this new column will be useful to use next in machine learning algorithms.

Also created a second hypothesis marking a value of 1 if the County is Dublin and 0 if not in a new feature whose name will be Dublin to use next as well.

Once again to handle the missing values opted to use the method “bfill” to fill the observation getting values from the next rows, once the methods “Median” and “Mean” were not working properly.

There are no duplicated observations, also no need to convert using “.astype()” for any features. Next, create a Correlation Matrix to understand the relationships of the features, once there are not as many features to be observed, also use a KDE plot to get the density of the Value feature.

In the Statistical Analysis steps, it was created a pair plot graphic, just to get the distribution of the Value feature; used a displot graphic to get the distribution of the Value of the house in Ireland, such as Mean, Median and Standard Deviation results of it, also check a distribution of Dublin where both are skewed to the right side.

Created some prints as statistical information to get some results of Mean, Median, Mode, Range, Standard Deviation, Variance, Interquartile Range, Percentile 25%, 50% and 75% and Coefficient of Variation for all Value houses in Ireland and just in Dublin.

Used Ordinal Encoder to encode the County names in a new feature whose name will be County\_Encoder, next checked if there are need to scale/transform features from the data, there is no need, but in case it is needed in the future it will create a new feature whose name is Value\_scaled to save the Feature Value scaled using “.scale” method from preprocessing in sklearn library.

Create a new data frame whose name is df\_outliers from df\_scaled to remove outliers using the IQR (Interquartile Range) Scoring technique to detect and remove any outliers. "instead look at individual features and identify extreme values in those features using interquartile range (IQR)" (Albon, 2018, p.85) in "df\_outliers".

Before dealing with the outliers there were 23491 observations and after 22703 in the total, less 788, next, used the boxplot, displot and individual boxplot to compare and check how the distribution is after the changes.

# **Machine Learning tecniquies**

The techniques chosen to be used to get results as the variation in the accuracy across three training splits in 20%, 25% and 30% are KNN Classification, Random Forest and Decision Trees.

# **KNN Classification**

First using KNN Classification and (k-Neighbors classification) “to make a prediction for a new data point, the algorithm finds the closest data points in the training dataset—its “nearest neighbors" (Müller and Guido, 2017, p.49), it was split into to two parts to train and test by a new data frame whose name is df\_ireland from df\_outliers, first with 20% of testing and 80% of training.

Generated a KNN Model with 3 neighbours using KNeighborsClassifier, and got the results: Accuracy: 0.88, Precision: 0.54 and Recall: 0.44 also tested with 25% of testing and 75% of training 30% of testing and 70% of training getting the respective results: 25% - Accuracy: 0.88, Precision: 0.55 and Recall: 0.44 and 30% - Accuracy: 0.87, Precision: 0.55 and Recall: 0.45. Next used a Confusion Matrix and classification report for more details.

Next use hyperparameters, to get the best results after training and testing the model, next it was used a plot to the KNN accuracy with varying numbers of neighbors "The plot shows the training and test set accuracy on the y-axis against the setting of n\_neighbors on the x-axis. While real-world plots are rarely very smooth, we can still recognize some of the characteristics of overfitting and underfitting (note that because considering fewer neighbors corresponds to a more complex model, the plot is horizontally flipped relative to the illustration in Figure 2-1)" (Müller and Guido, 2017, p.53).

Next used a k-neighbors regression, and the plot graphic to see k-neighbors regression with 1 neighbors and the other one with 5 neighbors. After it was using X and y to generate the data using make\_wave function, to "split the wave dataset into a training and a test set", "instantiate the model and set the number of neighbors to consider to 3" and "fit the model using the training data and training targets" (Müller and Guido, 2017, p.56)

Getting the following predictions: Test set predictions: [-0.05396539 0.35686046 1.13671923 -1.89415682 -1.13881398 -1.63113382 0.35686046 0.91241374 -0.44680446 -1.13881398] and Test set R^2: 0.83 which indicates a relatively good model fit.

Next plotted using a scatter plot to see the houses sold in Ireland and the Confusion Matrix also did the same to get the houses sold just in Dublin.

# **Random Forest**

Used also a random forest to collect decisions on trees, with RandomForestClassifier, in this case with 20% of testing and 80% of training got the results: Accuracy: 0.87, Precision: 0.54 and Recall: 0.40, with 25% of testing and 75% of training: Accuracy: 0.87, Precision: 0.52 and Recall: 0.40 and 30% of testing and 70% of training: Accuracy: 0.87, Precision: 0.55 and Recall: 0.40. Also used a Confusion Matrix and classification report for more details.

# **Decision Trees**

Used "Decision trees are widely used models for classification and regression tasks. Essen‐tially, they learn a hierarchy of if/else questions, leading to a decision." (Müller and Guido, 2017, p.84) using DecisionTreeClassifier, in this case with 20% of testing and 80% of training got the results: Accuracy: 0.87, Precision: 0.78 and Recall: 0.11, with 25% of testing and 75% of training: Accuracy: 0.88, Precision: 0.80 and Recall: 0.12 and 30% of testing and 70% of training: Accuracy: Accuracy: 0.87, Precision: 0.79 and Recall: 0.10. Also used a Confusion Matrix and classification report for more details, and used a plot\_tree to get a tree with max depth 3 as value.

# **Linear Regression**

Using linear regression to predict the value of the house, but instead using the df\_ireland and get all houses from the whole country, getting just df\_dublin for it, also created a new feature whose name is Sales just to test the Linear Regression algorithm in another perspective with random numbers from your choice, but remember to update the y split value, instead of 5 from the Sold feature, change to 9 from the Sales feature.

After using LinearRegression got a result of intercept: 0.07225566295545252 and coef: [-0.00164686 -0.00335004], also used a plot to visualise the actual and predicted values and another one showing the error terms of the predict in Sold feature, also obted the follow results: Mean\_Squared\_Error : 0.018913637109493044 and r\_square\_value: -0.005818644540288798, next used a scatter plot to see how the train, test and predict are.

# **Linear Regression Hypothesis**

Create a new feature whose name is Desired\_Value in the data frame df\_dublin to save a random percentage of prices to simulate a Desired Value Price for each property. Using numpy.random.uniform to get a random number between 0.5 to 0.25 "Draw samples from a uniform distribution." (NumPy Developers, 2024b).

Divided the data in train and test, next creating a function to calculate the mean values and Standard deviation after training the model, in this test got as result: CV Mean: 0.9677378837747778 and STD: 0.002413107279244489, also intercept and coefficient: -152.34604715945898 and [1.37592556] and created a new data frame whose name is df\_predict to save the actual price and predicted price.

# **K-Means**

Start using a histplot of the Frequency of Properties in Dublin per Neighborhood, next checked with a scatter graphic how the Acronym and Value\_scaled are in the distribution, and used the KMeans algorithms with 3 clusters to see how Acronym and Value\_scaled could be divided.

# **Cross-Validation and GridSearchCV**

Next used a cross-validation, importing cross\_val\_score and KFold from sklearn, splitting the data in X and y, creating a linear regression model, setting up k-fold cross-validation, performing cross-validation and getting R2 scores, using a for loop to print the R2 scores for each fold and next print the average R2 score across all folds.

Got as result:

* Fold 1: R2 Score = 0.0065503952365250084
* Fold 2: R2 Score = 0.011214924188233555
* Fold 3: R2 Score = 0.003982941438877674
* Fold 4: R2 Score = 0.0036515119062437984
* Fold 5: R2 Score = 0.009529201504550056
* Average R2 Score: 0.006985794854886018

Next using an example of GridSearchCV from the website analyticsvidhya.com, learning: Tune Hyperparameters with GridSearchCV.

"GridSearchCV acts as a valuable tool for identifying the optimal parameters for a machine learning model." (Shah, 2021)

"In GridSearchCV, along with Grid Search, cross-validation is also performed. Cross-Validation is used while training the model. As we know that before training the model with data, we divide the data into two parts – train data and test data. In cross-validation, the process divides the train data further into two parts – the train data and the validation data.

The most popular type of Cross-validation is K-fold Cross-Validation. It is an iterative process that divides the train data into k partitions. Each iteration keeps one partition for testing and the remaining k-1 partitions for training the model. The next iteration will set the next partition as test data and the remaining k-1 as train data and so on. In each iteration, it will record the performance of the model and at the end give the average of all the performance. Thus, it is also a time-consuming process." (Shah, 2021).

First, “Specifying Independent and Dependent Variables and splitting the data into train and test sets such as before, next Building Random Forest Classifier, Here, we created the object rfc of RandomForestClassifier(), Initializing GridSearchCV() object and fitting it with hyperparameters, and Getting the Best Hyperparameters.” (Shah, 2021). Resulted got:

* clf.best\_params\_: {'max\_depth': 12, 'max\_features': 12} and
* clf.best\_score\_: 0.8873466817307902

# **Conclusion**

After finishing all analyses, understanding the chosen data dealt with the EDA data preparation by finding unique values, replacing values, dropping irrelevant columns and renaming, handling missing values, creating hypotheses for machine learning algorithms, checking duplicated, type of conversion, relationships between the features, statical analyses, encoding values, scaled the numeric values and dealt with outliers, the data it was ready to use in Machine Learn models.

It was chosen some machine learning algorithms to tested such as KNN Classification, Random Forest and Decision Trees, once this data is not large enough to use PCA and try to predict with machine learning algorithms should be good to be used, the models chosen show us a good performance with good results in Accuracy, Precision and Recall, also used Linear Regression to predict values and next created one hypothesis to predict a get new house price. Also used K-Means with three clusters as a test and used Cross Validation with KFlod and GridSearch CV algorithms getting a good score as a result.

# **References**

Central Statistics Office (2024). Data Residential Property Price Index December 2023 - Central Statistics Office. [online] www.cso.ie. Available at: https://www.cso.ie/en/releasesandpublications/ep/p-rppi/residentialpropertypriceindexdecember2023/data/ [Accessed 16 Apr. 2024].

Cereniyim (2019). Distribution plot with mean and median. [online] GitHub Gist. Available at: https://gist.github.com/cereniyim/df1c8646411cc34eb9720a79400a200f [Accessed 14 Apr. 2024].

Data to Fish (2024). 5 ways to apply an IF condition in Pandas DataFrame - Data to Fish. [online] datatofish.com. Available at: https://datatofish.com/if-condition-in-pandas-dataframe/.

Müller, A.C. and Guido, S. (2017). Introduction to machine learning with Python : a guide for data scientists. Beijing: O’reilly.

NumPy Developers (2024a). numpy.random.randint — NumPy v1.20 Manual. [online] numpy.org. Available at: https://numpy.org/doc/stable/reference/random/generated/numpy.random.randint.html.

NumPy Developers (2024b). numpy.random.uniform — NumPy v1.22 Manual. [online] numpy.org. Available at: https://numpy.org/doc/stable/reference/random/generated/numpy.random.uniform.html.

Sanchhaya Education Private Limited, G. (2021). Select rows that contain specific text using Pandas. [online] GeeksforGeeks. Available at: https://www.geeksforgeeks.org/select-rows-that-contain-specific-text-using-pandas/ [Accessed 19 Apr. 2024].

Shah, R. (2021). GridSearchCV |Tune Hyperparameters with GridSearchCV. [online] Analytics Vidhya. Available at: https://www.analyticsvidhya.com/blog/2021/06/tune-hyperparameters-with-gridsearchcv/.

The Matplotlib Development Team (2024). matplotlib.pyplot.axvline — Matplotlib 3.8.4 documentation. [online] matplotlib.org. Available at: https://matplotlib.org/stable/api/\_as\_gen/matplotlib.pyplot.axvline.html#matplotlib.pyplot.axvline [Accessed 14 Apr. 2024].

The Pandas Development Team (2024a). pandas.DataFrame.isna — pandas 1.3.2 documentation. [online] pandas.pydata.org. Available at: https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.isna.html.

The Pandas Development Team (2024b). pandas.DataFrame.value\_counts — pandas 2.1.0 documentation. [online] pandas.pydata.org. Available at: https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.value\_counts.html#pandas.DataFrame.value\_counts.

The Pandas Development Team (2024c). pandas.isnull — pandas 2.2.2 documentation. [online] pandas.pydata.org. Available at: https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.isnull.html.

The Pandas Development Team (2024d). pandas.Series.str.split — pandas 1.3.0 documentation. [online] pandas.pydata.org. Available at: https://pandas.pydata.org/docs/reference/api/pandas.Series.str.split.html.