Predicting house prices using machine learning

Yuval Amar, Lin Sadon

Lecturer: Tammar Shrot

SCE - Sami Shamoon College of Engineering

Department of Software Engineering

Introduction:

The price of the apartment is the most important thing in a real estate transaction and controlling the pricing element is a key element in negotiations with private sellers and brokers. How can you really know what the price of the apartment is? The answer is not unequivocal, but there are a few parameters that determine the price.
In our project we are required to investigate the ability to predict real estate prices according to different characteristics of the property using different models from the field of artificial intelligence.
"Machine Learning is seeing its growth more rapidly in this decade. The use of machine learning (ML), which is a subfield of artificial intelligence (AI), to accurately forecast market prices using past data has proven successful".
Many applications and algorithms evolve in Machine Learning day to day. One such application is house price prediction.
House prices are increasing every year which has necessitated the modeling of house price prediction. These models, help the customers to purchase a house suitable for their need. by taking into consideration various parameters that affect the target value can predict the property cost.

In our project we took a dataset with 79 explanatory variables describing (almost) every aspect of residential homes in Ames, Iowa, to predict the final price of each home.
Our data set consists of many parameters, for example: the size of the apartment, the size of the yard, the size of the basement, the age of the house, location, number of floors in the house, number of rooms, pool, year of sale, etc.

The choice of models:

While our problem is not a classic problem of classification, i.e. for each line of parameters we do not need to say whether it is X, Y, Z, these for each line of parameters we need to give a result value in some range and therefore we chose to use both approaches: decision tree in regression and linear regression.

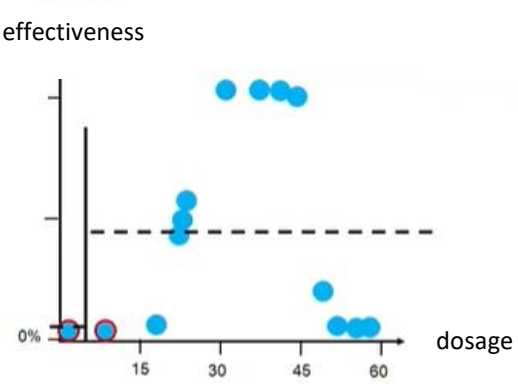
- regression decision tree:
  A regression decision tree in AI is a type of machine learning algorithm that is used for predicting a continuous target variable. It is a variation of the decision tree algorithm, which is typically used for classification problems.

A decision tree is built by recursively splitting the data on the feature that results in the largest decrease in impurity. Each internal node represents a feature, each branch represents a decision rule, and each leaf node represents the predicted outcome. The tree is built in a top-down, greedy manner, by selecting the feature that results in the most homogeneous subsets of the data at each step. Regression decision trees can handle both linear and non-linear relationships between the input features and the target variable, and they can also handle interaction between features. They are also able to handle missing values, and they are easy to interpret and visualize.

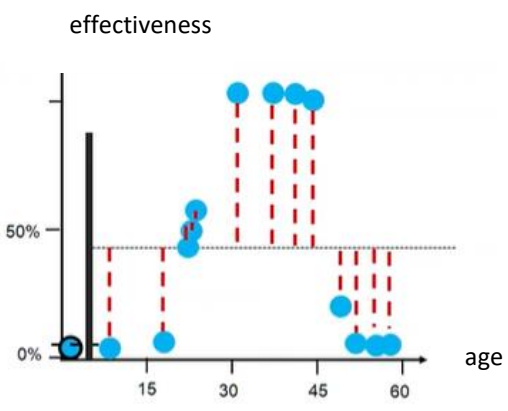
In the case of regression decision tree, the outcome is a continuous variable and the output is a real number.

We will present an example that shows the process of building a decision tree in regression with 2 parameters:

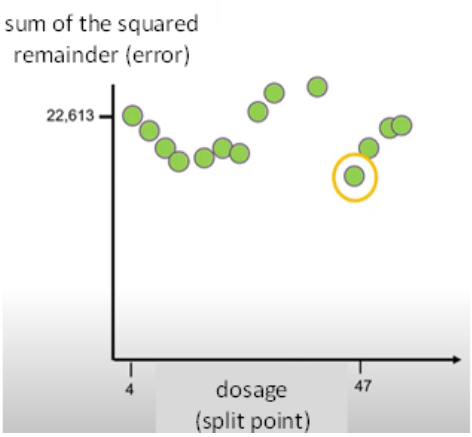
Suppose we want to predict the effectiveness of a drug against age and dose. For each parameter will be built a graph that showing the parameter versus effectiveness, we will measure the average between every 2 points[1], we will divide the graph according to the average we received, for each part we will find the average of the effectiveness and this will be the forecast of the tree for each group. After that we will measure the distance between what the tree predicts and the real effectiveness, we will square it and sum all the distances and let's call it the error[2], we will repeat the process with all the points in the parameter. We choose the point with the lowest error[3]. We will repeat the process with the rest of our parameters and choose the parameter that provided us the lowest error. Finally, we will repeat the process until we complete the whole tree[4].



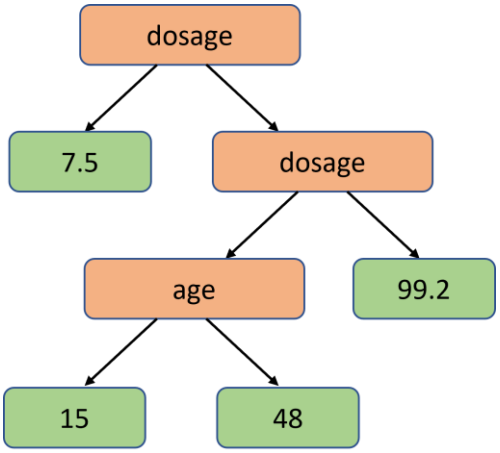
[1]



[2]



[3]



[4]

- <u>Linear regression:</u>

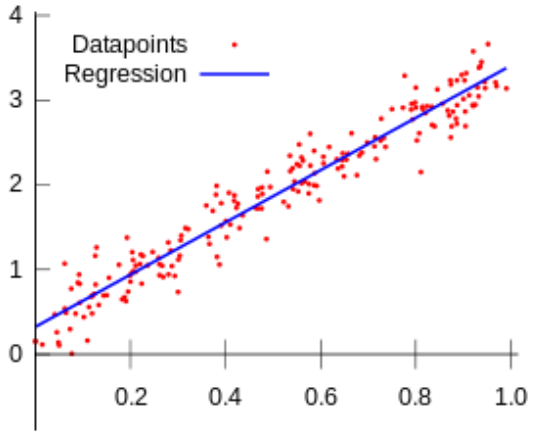
  Linear regression in AI is a statistical method and a supervised learning algorithm that is used for predicting a continuous target variable. It is a linear approach to model the relationship between a dependent variable (y) and one or more independent variables (x). The goal is to find the best fit line that minimizes the sum of the squared differences between the predicted and actual values of the dependent variable. Linear regression assumes a linear relationship between the input features and the target variable, and it is used to model continuous variables. The model is represented by an equation of the form:

  y = b0 + b1x1 + b2x2 + ... + bn*xn

  where y is the target variable, x1, x2, ..., xn are the input features, b0 is the y-intercept, and b1, b2, ..., bn are the coefficients of the independent variables. These coefficients are estimated from the data using a method called least squares. Linear regression is a simple and interpretable algorithm, it's easy to implement and computationally efficient. However, it has a few limitations, such as it only works well when the relationship between the input features and the target variable is linear and also it's assume that the errors are normally distributed and have constant variance.

  In the case of linear regression in AI, the input can be multiple variables, and the output is a real number.

We will present an example of how the linear line on which the result is based is represented.



In this case we will present an example in two dimensions for 2 parameters - the variable parameter (X axis) and the result parameter (Y axis).
In a similar way, when the number of parameters will be 3 the linear line will be calculated for a 3D graph and so on.
When we get a certain value we can place it in the linear equation and get the predicted result.

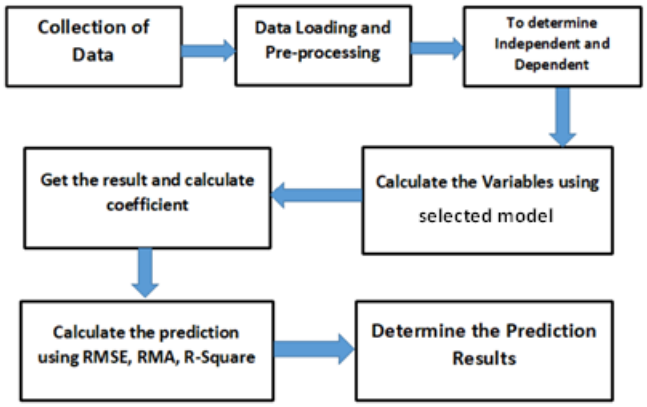
Models:

We used a library Scikit-learn, also known as sklearn, is an open-source Python library for machine learning. It provides a variety of tools and algorithms for data analysis and modeling, including classification, regression, clustering, dimensionality reduction, model selection, and preprocessing.
sklearn provides a consistent and easy-to-use interface for working with machine learning algorithms. Scikit-learn is widely used in data science projects, it's simple to use, it's well documented, it's efficient and it's a great tool for data scientists, machine learning engineers and researchers who want to quickly train models and run experiments.

From the library we used two main models:
sklearn.linear_model.LinearRegression, sklearn.tree.DecisionTreeRegressor.

System overview:

Link to the project - https://github.com/yuvalamar1/Predict-house-prices-AI



System overview for - regression decision tree:

1. Reading the database from a CSV file.

2. Analysis of the information obtained from the data set.

3. Completing missing information in the data set according to statistical analysis.

4. Cleaning the information, removing irrelevant information that may harm the quality of the model's result.

5. Coding relevant fields in the data set (to numerical values).

6. Adapting the data set to a data set that knows how to work with the existing model (data set for - sklearn.tree.DecisionTreeRegressor).

7. We will divide the model into parameters for training and for testing X_train, X_test, Y_train, Y_test (80% of the dataset for training and 20% for testing).

8. We will train the model (using the scikit-learn API).

9. After training the model, predict the result for the test dataset.

10. Calculate and present the MSE (MEAN SQUARED ERROR)

11. Calculate and present the R2 SCORE.

12. Displaying the running time of the model training.

System overview for - Linear regression:

1. Reading the database from a CSV file.

2. Analysis of the information obtained from the data set.

3. Completing missing information in the data set according to statistical analysis.

4. Cleaning the information, removing irrelevant information that may harm the quality of the model's result.

5. Coding relevant fields in the data set (to numerical values).

6. Adapting the data set to a data set that knows how to work with the existing model (data set for - sklearn.linear_model.LinearRegression).

7. Extract the result column from the dataset.

8. We will divide the model into parameters for training and for testing X_train, X_test, Y_train, Y_test (80% of the dataset for training and 20% for testing).

9. We will train the model (using the scikit-learn API).

10. After training the model, predict the result for the test dataset.

11. Calculate and present the MSE (MEAN SQUARED ERROR)

12. Calculate and present the R2 SCORE.

13. Displaying the running time of the model training.

14. Presentation of a graph showing a comparison between the results predicted by the model and the real results.

Dilemmas and challenges during work:

1. Searching for a data set with many characteristics in order to predict the result well.

2. Analysis of the information - as part of the preliminary process for training the model, we were asked to do statistical analyzes in order to clean up information and understand which characteristics have a good effect on the result.

3. Coding the information, as we mentioned, part of the process was to convert the information from an existing data set to one that knew how to work with the model.

4. Technical problem - a MacBook computer with a MacOS operating system does not support the use of the necessary libraries, therefore part of the work was done in jupyter notebook.

5. Technical problem - at first we chose a data set with 20,000 rows which required a particularly powerful computer (we were afraid we would have to limit the depth of the decision tree) and a long model training time but finally we chose a smaller data set.

The data set:

Field	Description
MSSubClass	Identifies the type of dwelling involved in the sale
MSZoning	Identifies the general zoning classification of the sale
LotFrontage	Linear feet of street connected to property
.	.
.	.
.	.
PoolArea	Pool area in square feet
YrSold	Year Sold (YYYY)
LotArea	Lot size in square feet
SaleCondition	Condition of sale

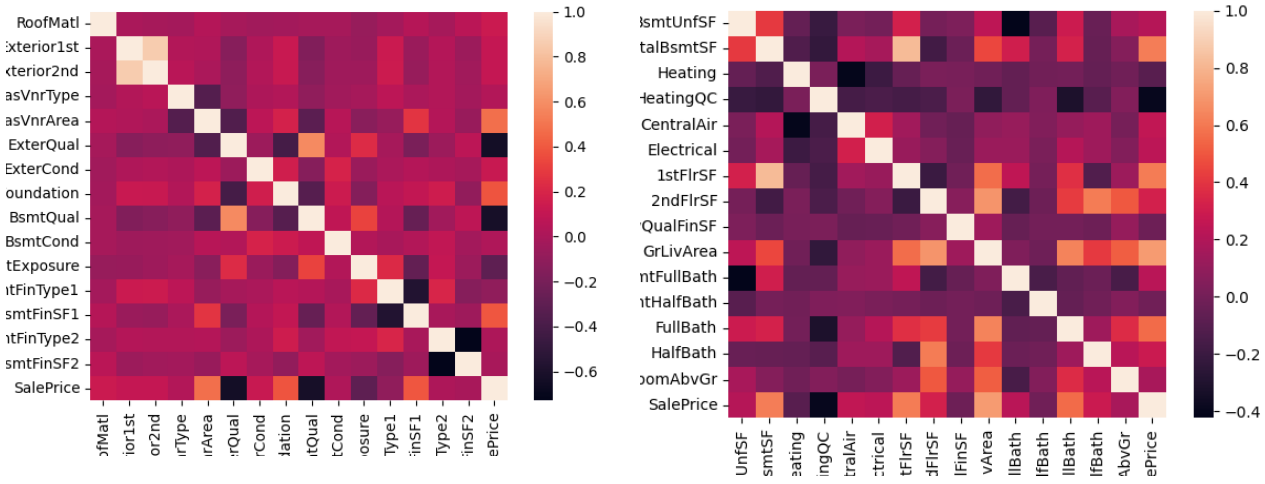
Our data set is represented by 1460 rows and 80 features that represent purchase history, since this information is real information from past sales, so there are many parameters that are not perfect, therefore as part of the information cleaning process we will present the steps in cleaning the information.
It is important to note that this process happens for the use of both algorithms.

Data analysis:

In order to get the best prediction result, we would like to do some preliminary research on the data set we received. This process is an important process in order to understand what are the features that most strongly influence our result column and which algorithm should be chosen for machine learning.

We used the "seaborn" and "matplotlib" libraries that exist in the language in order to analyze the information.

We wanted to show correlations between variables in order to clean features with a bad correlation, we chose to present the information using heat maps between the features.



Search for missing features - we will look at the data set and it will appear that there are features that have not been fully filled in the table, so we will decide to download them in order not to confuse the model.

Search for missing fields - we will search on the train's table for places where fields are missing, i.e. columns where they forgot to fill in the field, in order not to damage the prediction of the model we will fill in the missing fields with the median value of that characteristic.

Coding the table fields - the field types of our data set consist of different types, integers, real numbers and objects, because we are working with an existing SCIKIT model we must code the fields in the table each attribute will be represented by numbers. Let's see an example for one feature coding (MSZoning).

the original value	The meaning of the value	The value after encoding
A	Agriculture	5
C	Commercial	0
FV	Floating Village Residential	1
I	Industrial	6
RH	Residential High Density	2
RL	Residential Low Density	3
RP	Residential Low Density Park	7
RM	Residential Medium Density	4

The results of the models :

We will present the nature of the results of the models with the help of two characteristics MSE and R2 score.

MSE (Mean Squared Error)- is a commonly used evaluation metric for regression problems. It represents the average squared difference between the predicted values and the actual values. A low MSE indicates that the model is doing a good job of predicting the target variable, as the differences between the predicted and actual values are small. On the other hand, a high MSE indicates that the model is not doing a good job of predicting the target variable, as the differences between the predicted and actual values are large.

Mean squared error	$\text{MSE} = \frac{1}{n}\sum_{t=1}^{n} e_t^2$

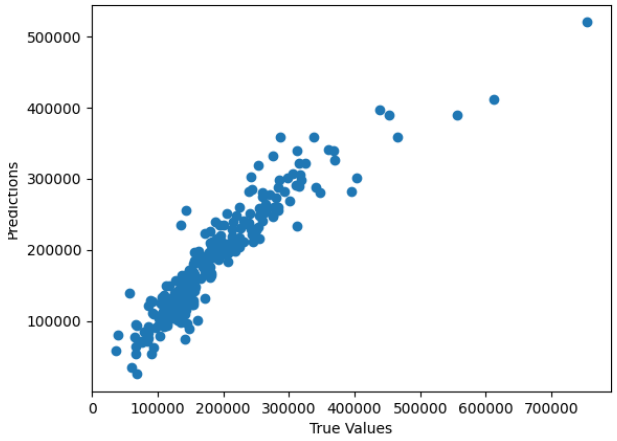
The R-squared (R2) score is a statistical measure of how well the regression predictions approximate the real data points. It ranges between 0 and 1, where a score of 1 means the predictions perfectly fit the data, and a score of 0 means the predictions do not fit the data at all. In simple terms, R-squared tells you how much of the variance in the target variable (y) is explained by the variance in the feature variable (x). It also tells you how well the model fits the data.

Results for a linear regression:

We will present the result by a graph showing a comparison between the result that the model expected versus the real results, in practice the model produces a linear line on a graph with 80 dimensions (as the number of features in the data set) and when it receives new information it places the information on the same linear line and receives the predicted result.

We will not be able to show this graph due to the number of properties in the table.

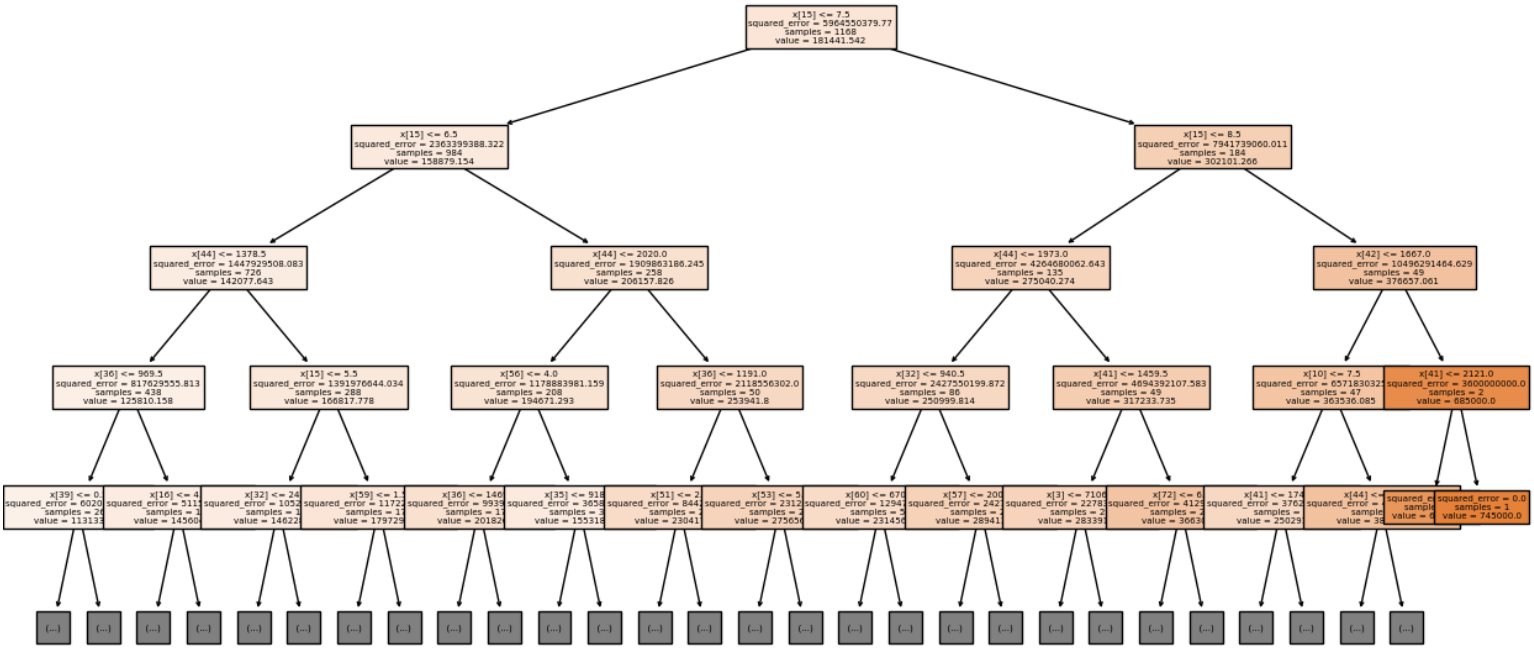
```
Mean Squared Error:  1152145912.9688354
R-Squared:  0.8497917973958826
```



Results for a regression decision tree:

We will present the decision tree for a depth of 5, but in practice the depth of the decision tree obtained after training the model is 24.

```
Mean Squared Error:  1677876552.5342467
R-Squared:  0.7812509524090292
Maximum depth of the tree:  24
```



Conclusions and summary

Since we used an existing library that builds the model by itself, the main work in the project was to understand how the model works and how to enter a data set into the model that would give the best result.
During the work on the project, we realized that the part of cleaning data is a very important step.
The parameters for comparing the models are the MSE and R2 SCORE (as we detailed earlier). The use of the linear regression model gave the results MSE=1152145912 and R2 SCORE=0.849, on the other hand, the use of the regression decision tree model gave the results MSE=1677876552 and R2 SCORE=0.781. The linear regression model presented better results than a regression decision tree model, meaning it predicted the house price closer to reality.
It is important to note that there is an XGBOOST algorithm that predicts the house price more accurately with MSE=426692527 and R2 SCORE=0.913.
XGBoost uses decision trees as the base model, and it trains new trees on the residuals of the previous tree, this allows the model to learn from the mistakes of the previous tree and improve the predictions.

Decision Tree Regression is a powerful algorithm that can handle non-linear relationships and interactions between features in a dataset.
Decision Tree Regression can handle categorical variables and missing values directly, while linear regression requires these variables to be transformed or encoded.
Linear Regression is a simple and interpretable algorithm that is best suited for linear relationships. It is particularly useful when the goal is to understand the relationship between the features and the target variable.
Linear regression is sensitive to outliers and can be affected by the presence of poorly correlated features. To address this, techniques like data cleaning, removing outliers, or using regularization can be used.
Linear Regression has the advantage of being a fast algorithm and it's easy to interpret the results, while Decision Tree Regression could be slow especially with large datasets.

The project helped us deeply understand the world of machine learning and the various supervised learning algorithms. In addition, it showed us how much the field of AI is developing and in order to solve problems with the help of AI we do not need to clearly define the problem or the database.

Bibliography:

https://www.kaggle.com/competitions/house-prices-advanced-regression-techniques/overview

https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeRegressor.html

https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html

https://www.mecs-press.org/ijieeb/ijieeb-v12-n2/IJIEEB-V12-N2-3.pdf

https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8697639

https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8882834