A picture containing logo

Description automatically generated

Principles Of Data Science

10204280

Section (4)

Submitted To:

Dr. Murad Yaghi

Submitted By:

Nasrullah Rami Alhaj Hamad

1.1: Types of Machine Learning:

|  |  |  |
| --- | --- | --- |
| **###################**  **###################**  **###################** | **Supervised Learning** | **Unsupervised Learning** |
| **Definition** | Mapping inputs and outputs learned and trained from existing inputs and outputs. | Maps inputs and outputs on unknown patterns without human supervision. |
| **Applications (use)** | Spam detection, the relation between house prices | employee personalities, medical imaging. |
| **Strengths** | Takes direct feedback to check if the prediction is correct. | This learning will learn the data itself without a need to provide it |
| **Limitations** | The labels of inputs and outputs need to be carefully trained and tested, and it requires a lot of time. | Complex when producing intended outcomes.  Produces inaccurate results unless a human intervening |
| **Common Algorithms** | Classification  Regression | Clustering.  Association |

1.2: Types of Supervised Learning:

|  |  |  |
| --- | --- | --- |
| **#####################** | **Classification** | **Regression** |
| **Type of Learning** | Supervised Learning | Supervised Learning |
| **Applications (use)** | Discrete values:  Positive or negative  Email is spam or not. | Continuous values:  Price  Body Weight |
| **Strengths** | Classifications are good at showing the probabilistic visuals of the data. | Regressions are straightforward and easy to understand. And they can easily be updated |
| **Limitations** | It is based on existing corporal features. | Estimation of values can lead to inaccurate results |
| **Common Algorithms** | KNN  Decision Tree | Linear Regression  Decision Tree Regressor |

1.3 Computing Systems:

AWS and Azure are cloud computing services that come up with a range of services and other computing resources while using hardware and infrastructure to supply these services.

**Structure:**

AWS’s structure and Azure’s use a distributed architecture, they both use a variety of physical and virtual servers, as well as storage devices, to provide their service.

**CPU and GPU:**

AWS offers a variety of CPU and GPU configurations depending on the different occasion, if, for example, the case was with high-performance processes, AWS offers Intel Xeon and AMD EPYC, and for GPU, AWS offers NVIDIA Tesla and AMD Radeon.

Azure also gives a variety of CPU and GPU options, such as Intel Xeon, AMD EPYC, and NVIDIA GPU.

**Storage:**

AWS has many storage options, for block storage EBS, file storage EFS, and object storage S3. And it has a service to manage databases and other data such as AWS RDS.

Azure also has storage options depending on the storage type, for block storage Azure Disk Storage is used. For file storage, Azure uses Azure File Storage. For object storage, Azure Blob Storage is used and uses Azure SQL Database as a service for managing data and databases.

**Hardware Aspects:**

AWS and Azure have a range of software services for analytics, AI, machine learning, and more.

**Conclusion:**

AWS and Azure both provide a set of computing services. The choice of choosing either of them will depend on the use occasion and the requirements.

Part 2.1: The data lifecycle stages:

|  |  |
| --- | --- |
| **The Data Life Cycle Stage** | **The importance of the given Stage in your project lifecycle** |
| Understanding the problem | You must be aware of the problem you’re solving and figure it out, so you start gathering data to solve it |
| Gathering Data | Gather useful information from the sources around, with Python, Web APIs, downloaded files, texts, or CSV files. |
| Cleaning Data | Cleaning data is done by converting data into a different format, to begin processing and analyzing, it includes replacing values or filling missing values with suitable data. |
| Exploring Data | Data now must be examined because different data such as nominal, ordinal, categorical, and numerical data require different handling. |
| Feature Engineering | It’s good to minimize the dimension of the data set, filter out correlated data, and allow us to deduce the change in column values that would affect other columns. |
| Modeling Data | The most interesting part of the cycle, models are used to differentiate different classes and predict a specific event by using different types of models with each having its advantages and drawbacks. Many evaluation measures are needed to calculate the performance of the model, or by using advanced techniques like the confusion matrix and its terms. |
| Interpreting Data | The most essential part of the cycle, depends on the accuracy of the predicting model, in this stage, the data must be presented to a human being, and it is usually delivered in a form of results that can be achieved through proper visualization. |

The interpreting stage requires some visual plots that feature all the columns inside the dataframes. To do that, I separated all the columns into categorical and numerical data. And I used a histogram to plot all the numerical data and a pie chart to plot all the categorical data.

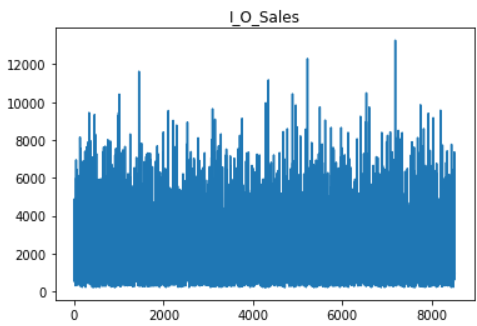
Here’s an example for the histograms used for some of the columns:

Chart

Description automatically generatedChart, histogram

Description automatically generated

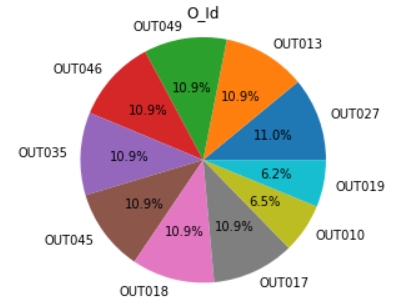
Chart, histogram

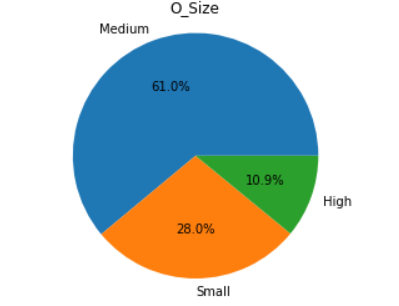
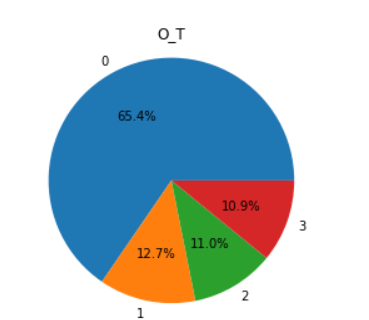
Description automatically generated

And here are some examples for the plotting of the categorical data using Pie Chart:

Chart, pie chart

Description automatically generated





These charts included some features after being preprocessed.

**Part 2.2:**

|  |  |  |
| --- | --- | --- |
| **Step** | **data pre-processing step** | **Description** |
|  | I\_Fat\_C | Reduced its unique values to two only, LF = Low Fat & reg = Regular |
|  | I\_W | Filled with mean |
|  | O\_Size | Filled with the mode |
|  | O\_Establ\_Y | Changed its data type to object |
|  | I\_Vis | Replaced 0 with NaN, and then filled missing values with the mean |
|  | I\_Id | Dropped the column |
|  | O\_T | I changed its values to numbers in a certain order |
|  | I\_Recalled | Replaced No with 0, Yes with 1 |

**2.3:**

First, I made the I\_Fat\_C column hold only two unique values instead of 5, because other values held the same value but with small letters. So, they had to be changed to make sense for the model, so it predicts data better.

I filled in the missing values in I\_W with its mean, because the numbers were suitable and close to the mean.

I filled in the missing values in the column O\_Size with its mode, to make sense for the model and therefore predict better results.

I changed the data type of O\_Establ\_Y to object because it is ordinal and I want to use it with the other ordinal data when doing the encoding step.

In I\_Vis, I replaced the zeros with NaN, so I can fill it with a better value, so it makes sense to the model, I filled the missing values with the mean afterward.

I dropped the I\_Id column because it isn’t going to be used in the predictions.

I rearranged the values in the O\_T column to numbers, so each unique value holds a number since it is nominal.

I replaced each value that holds No with 0 and 1 to each value that holds Yes, to make it make sense to the model and for it to be easier to predict.

I made sure to preprocess all the data in the features gotten from the dataframes, to ensure the data is clean and ready to be trained to the model, to get better results.

**3.1:** Code

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Alpha (LR)** | **R2** | **MSE** | **RMSE** | **MAE** |
| **0.00001** | 30.2 | 1954812 | 1398 | 1049 |
| **0.000001** | 29.5 | 1975841 | 1405 | 1062 |
| **0.00000001** | 0.02 | 2738115 | 1654 | 1179 |
| **K value** | **Accuracy** | **Precision** | **Recall** | **F1-Score** |
| **3** | 0.72 | 0.54 | 0.52 | 0.52 |
| **5** | 0.75 | 0.56 | 0.51 | 0.51 |
| **11** | 0.76 | 0.55 | 0.50 | 0.49 |

**3.2:** Code

**3.3:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Approach (Linear Regression)** | **R2** | **MSE** | **RMSE** | **MAE** |
| without normalization | 30.2 | 1954812 | 1398 | 1049 |
| Min-max normalization | 0.56 | 0.007 | 0.08 | 0.06 |
| Z score Normalization | 0.56 | 0.41 | 0.64 | 0.47 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Approach (KNN)** | **Accuracy** | **Precision** | **Recall** | **F1-Score** |
| without normalization | 0.76 | 0.55 | 0.50 | 0.49 |
| Min-max normalization | 0.76 | 0.55 | 0.50 | 0.49 |
| Z score Normalization | 0.96 | 0.97 | 0.94 | 0.95 |

**3.4:**

Using the linear regression model, my highest r2 value was 30.2 and the mean square error was 1954812, the root mean square error was 1398, and the mean absolute error was 1049, I changed the learning rate to get different values, but these were my highest values.

For KNN I was using different k-values each time I train the model, I got my highest performance when the k-value was equal to 11, I got an accuracy of 76%, and precision of 55%, and a recall of 50%, and f1-score was 49%.

I used these models without normalization first, but with normalization, I got a lower r2 score for the linear regression, but the other measures were better with normalization, for KNN using the min-max normalization, the results did not change, but with the z-score normalization, they rose to 99% which is very high based on the previous results.

3.5:

When I started using the linear regression model, I made the learning rate equal to 0.1, but all the answers gave me null, so I started changing it by adding more zeros and it started giving me more reasonable numbers until I got the learning rate 0.0000001 when it started to give me a reasonable r2 value. But the other linear regression evaluation measures didn’t make sense and I couldn’t fix it, so I just left them as they are, but with using the normalization methods, the values I got were reasonable, since they gave me the best values of the measures of the linear regression, so I chose them. The z-score normalization gave me better MSE, RMSE, and MAE values. Since they were closer to 0

When I first programmed the KNN classification model, I made the k value equal to 3, and it gave me accuracy equal to 99.9 which I wondered if it was true, then I changed the preprocessing, and it gave me a more reasonable accuracy which was 78. The other evaluation measures like the precision, recall, and f1 score gave me values between 50 and 60, which I included in the report, and then I changed the k values to 5 and 11 and it the values started getting better. When I used the k value = 11, it gave me the best measures in the KNN model. After that, I used the normalizations, and the min-max normalization gave me the same values as without normalization. But the z-score normalization gave me the best measures yet, the accuracy went to 96 and the other measures were around that. So, I wrote it in the report too.

In conclusion, the KNN model gave me a better performance than the linear regression, and that is because the label was better used for classification than regression. And the normalizations for the Linear regression gave me more logical measures for the MSE, RMSE, and MAE. I would recommend companies use the KNN classification model for these predictions because it would give a better performance than linear regression.

References:

* EdPrice-MSFT (n.d.). AWS to Azure services comparison - Azure Architecture Center. [online] learn.microsoft.com. Available at: <https://learn.microsoft.com/en-us/azure/architecture/aws-professional/services>.