PEOPLE'S DEMOCRATIC REPUBLIC OF ALGERIA

Ministry of Higher Education and Scientific Research

UNIVERSITY OF ECHAHID HAMMA LAKHDAR - EL OUED

FACULTY OF EXACT SCIENCES

Computer Science Department

Thesis

Submitted in partial fulfilment of the requirements for the degree of

Field: **Mathematics and Computer Science**

Specialty: **Computer Science**

Title

Enhancing the optical correction with artificial intelligence

Presented by:

- **Abbas Assia**
- **Ferdia Chaima**
- **Mahcene Wijden**

supervisor:

Abbas Messaoud

Academic year: 2023 - 2024

بْسمِ ـَالله الرحمِن الرحيِم

الإهـداء

بْسمِ ـَالله الرحمِن الرحيِم " وقل اعملوا فسيرى الله عملكم ورسوله والمؤمنون"

إلهي لا يطيب الليل إلا بشكرك، ولا يطيب النهار إلا بطاعتك، ولا تطيب اللحظات إلا بذكرك، ولا تطيب الآخرة إلا بعفوك، ولا تطيب الجنة إلا برؤيتك جل جلالك.

إلى من بلغ الرسالة وأدى الأمانة ...ونصح الأمة نبي الرحمة ونور العالمين سيدنا محمد ﷺ

نحمد الله عز وجل لإتمام هذا البحث، وإلى الذي وهبنا كل ما يملك حتى نحقق له آماله. إلى من كان يدفعنا قدما نحو الأمام لنيل المبتغى، إلى الإنسان الذي امتلك الإنسانية بكل قوة، إلى الذي سهر على تعليمنا بتضحيات جسام مترجمة في تقديسه للعلم، آبائنا الأعزاء على قلوبنا، إلى مدرستنا الأولى في الحياة التي رعتنا، إلى التي صبرت على كل شيء، إلى التي وهبت فلذة كبدها كل العطاء والحنان حق الرعاية وكانت سندنا في الشدائد، وكانت دعواها لنا بالتوفيق، تتبعتنا خطوة بخطوة في عملنا، إلى من ارتحنا كلما تذكرنا ابتسامتها في وجهنا نبع الحنان أمهاتنا أعز ملاك على القلب والعين جزاها الله عنا خير الجزاء في الدارين، إليهما نهدي هذا العمل المتواضع لكي ندخل على قلبهما شيئا من السعادة. إلى إخوتنا وأخواتنا الذين تقاسموا معنا عبء الحياة، كما نهدي ثمرة جهدنا لأستاذنا الكريم عباس مسعود وإلى كل أساتذة قسم الإعلام الآلي، إلى كل من ساهم في هذا العمل من بعيد أو قريب وإلى كل من يؤمن بأن بذور نجاح التغيير هي في ذواتنا وفي أنفسنا قبل أن تكون في أشياء أخرى.

قال الله تعالى " إن الله لا يغير ما بقِومٍ حتى يغيروا ما بأنفسهم"

سورة الرعد. الآية 11

ملخص

يعتبر دمج الذكاء الاصطناعي والعلوم الطبية قفزة حيوية في الرعاية الصحية الحديثة، محدثاً ثورة في الأساليب التقليدية. من خلال تحليل البيانات الطبية المعقدة، تمكّن خوارزميات الذكاء الاصطناعي من تشخيصات دقيقة، وتنبؤ مبكّر بتطوّر المرض، مما يعزز الرعاية ونتائج المرضى. في هذه المذكرة، نستكشف آثار التعلم الآلي في مجال الرعاية الصحية، مع التركيز بشكل خاص على طب العيون وتصحيح الرؤية. من خلال تحليل مجموعات بيانات شاملة، يساعد التعلم الآلي في تشخيص حالات العين، وتنبؤ تطور المرض، ووضع خطط علاج مخصصة. تهدف جهودنا إلى مساعدة الأطباء في مستشفى العيون بالوادي في الحصول على قياسات دقيقة لوصفات النظارات الطبية. ولهذا الغرض، قمنا بتجميع مجموعة بيانات شاملة وتطبيق خوارزميات انحدار تراجع التعلم الآلي للتنبؤ بالقياسات بدقة.

الكلمات المفتاحية: الذكاء الاصطناعي، العلوم الطبية، خوارزميات الذكاء الاصطناعي، التعلم الآلي، تحليل البيانات، طب العيون، تصحيح الرؤية، التنبؤ بالأمراض، وصفات النظارات الطبية.

Abstract

The integration of artificial intelligence with medical sciences represents a vital leap forward in modern healthcare, sparking a revolution in traditional methodologies. Through the analysis of complex medical data, AI algorithms enable precise diagnoses and early disease prediction, thus enhancing patient care and outcomes. In this memorandum, we explore the impact of artificial intelligence in healthcare, with a specific focus on ophthalmology and vision correction. Through the analysis of comprehensive datasets, AI aids in diagnosing eye conditions, predicting disease progression, and devising personalized treatment plans. Our efforts aim to assist physicians at Kouba Hospital in obtaining accurate measurements for medical glasses prescriptions. To this end, we have compiled a comprehensive dataset and applied machine learning regression algorithms for precise measurement prediction.

Keywords: Artificial intelligence, Medical sciences, AI algorithms, Machine learning, Data analysis, Ophthalmology, Vision correction, Disease prediction, Medical glasses prescriptions.

Résumé

L'intégration de l'intelligence artificielle avec les sciences médicales représente un pas en avant vital dans les soins de santé modernes, suscitant une révolution dans les méthodologies traditionnelles. Grâce à l'analyse de données médicales complexes, les algorithmes d'IA permettent des diagnostics précis et une prédiction précoce des maladies, améliorant ainsi les soins et les résultats pour les patients. Dans cette note, nous explorons l'impact de l'intelligence artificielle dans le domaine de la santé, avec un accent particulier sur l'ophtalmologie et la correction de la vision. À travers l'analyse de jeux de données complets, l'IA aide à diagnostiquer les affections oculaires, prédire la progression des maladies et élaborer des plans de traitement personnalisés. Nos efforts visent à aider les médecins de l'Hôpital des yeux d'Eloued à obtenir des mesures précises pour les ordonnances de lunettes médicales. À cette fin, nous avons compilé un ensemble de données complet et appliqué des algorithmes de régression d'apprentissage automatique pour prédire avec précision les mesures.

Mots clés: Intelligence artificielle, Sciences médicales, Algorithmes d'IA, Apprentissage automatique, Analyse de données, Ophtalmologie, Correction de la vision, Prédiction des maladies, Prescriptions de lunettes médicales.

Contents

Introduction

The combination of machine learning and medical science is a significant step forward in modern healthcare. This collaboration not only uses advanced computing methods but also has the potential to completely change traditional medical practices. Machine learning, a part of artificial intelligence, allows computers to learn from large sets of data and make predictions or decisions without specific programming. This opens up new possibilities for innovation in medicine.

In recent years, machine learning has become an essential tool in healthcare. It helps with various aspects, such as diagnosing diseases, planning treatments, and caring for patients. By examining complex medical data, like electronic health records, medical images, and genetic information, machine learning algorithms can find hidden patterns and connections. This leads to more accurate diagnoses and personalised treatment plans. Additionally, machine learning helps predict how diseases progress and how patients respond to treatment. This means doctors can intervene earlier and improve patient outcomes.

In this memorandum, we explore how machine learning is transforming healthcare, with a focus on ophthalmology and vision correction. By analysing large datasets, machine learning helps diagnose eye conditions, predict disease progression, and guide personalised treatment plans.

More specifically, our work aims to help doctors at the eye hospital in El Oued to make accurate medical glasses measurements. Thus, we have collected the necessary dataset and applied machine learning regression algorithms to predict correct measurements. The remainder of this document is organised as follows:

- In the first chapter scope of the study

- In the second chapter The Fundamentals of Machine Learning
- In the third chapter highlights our dataset collection procedure
- In the fourth chapter describer our regression models for predicting eye glasses measurements

Chapter 1

Scope of the study

1.1 The hospital

The hospital specializing in eye diseases in El Oued (known as Kouba Hospital) was established on 24/08/2014 through cooperation between Algeria and Kouba. The hospital is a polar and regional ophthalmologic centre, equipped with state-of-the-art technological machines and apparatus managed by specialised doctors and technicians. Hundreds of patients visit the hospital daily. According to statistical results from 2022, about 300 patients are seen each day.

1.2 Problematic

Our study focuses on the division responsible for eye correction for medical glasses. The measurement specifications in this division show that a large number of patients require this service each day. However, the responsible optician can only handle a limited number of patients daily. Although they use a sophisticated machine for measuring parameters, the optician must finalise the operation manually. To address this situation, the use of AI techniques has been considered to accelerate patient treatment.

1.3 Proposed solution

Utilising machine learning is the best suggestion to solve the presented problems. This involves collecting a dataset that includes the measurements and diagnostics from the Nidek machine along with the manual corrections made by specialists. Then, a machine learning model, such as a linear regression model, can be trained on the obtained data. After evaluating the model's performance, it can be used practically by inputting the dataset from the machine to obtain the corrected measurements.

Chapter 2

Fundamentals of machine learning

In traditional computing, algorithms are sets of explicitly programmed instructions used by computers to calculate or problem-solve, but with very large datasets, it's difficult to solve some of these problems with traditional programming, so we're going to choose artificial intelligence, especially machine learning. Thus, as stated in (Hands on machine learning): "Machine Learning shines is for problems that either are too complex for traditional approaches or have no known algorithm ".

In this section, we will present different types of machine learning. We will focus more on regression models, as they are concerned by our study.

Note that most content of this section is extracted from the book: "Hands-on Machine Learning (Aurélien Géron)" [5]

2.1 Types of Machine learning

There are so many different types of machine learning systems, of which we can summarize as follow:

2.1.1 Supervised Learning

In supervised learning, the training dataset you feed to the algorithm includes the desired solutions, called labels.

Two categories of algorithms come under the umbrella of supervised learning:

- Classification: The spam filter is a good example of this, it is trained with many example emails along with their class (spam or ham), and it must learn how to classify new emails. Classification: The spam filter is a good example of this, it is trained with many example emails along with their class (spam or ham), and it must learn how to classify new emails.
- Another typical task is to predict a target numeric value, such as the price of a car, given a set of features (mileage, age, brand, etc.) called predictors. This sort of task is called regression to train the system, you need to give it many examples of cars, including both their predictors and their labels.

2.1.2 Unsupervised Learning

In unsupervised learning, the training dataset is unlabeled. The system tries to learn without a teacher. For example, clustering algorithms can help to recommend items to users based on the purchasing or viewing patterns of similar users, improving the personalization of recommendations.

2.1.3 Reinforcement learning

Reinforcement Learning is a very different beast. The learning system, called an agent in this context, can observe the environment, select and perform actions, and get rewards in return. As an example, self-driving cars, where the agent learns to navigate the traffic and road conditions by receiving feedback from sensors and cameras.

2.2 Regression

A regression analysis problem is used when the target variable is a real or continuous value. It examines a mathematical formula that describes the relationship between the independent variable(s) and the target variable.[7]

2.2.1 The process of building a Regression model

To generate a robust Regression model that meets our requirements, we should follow the following steps [1]

1. **Data Collection**

   In this phase, relevant data is gathered from various sources to train the Regression Model and enable it to make accurate predictions.

2. **Preprocessing and Preparing the Data**

   Preprocessing and preparing data are an important step. It consists of transforming data into a format that is suitable for training and testing models. So, it enables cleaning data, filling missing data, scaling data,etc.

3. **Selecting the Right Regression Model**

   Selecting the right algorithm plays a pivotal role in building an accurate and successful model, with the presence of numerous algorithms and techniques:

   (a) **Linear Regression:** is a means of regression analysis used widely in the fields of mathematics and data science. In machine learning, it's often used to find the optimal linear model of two parameters or datasets.[10]
   see figure2.1

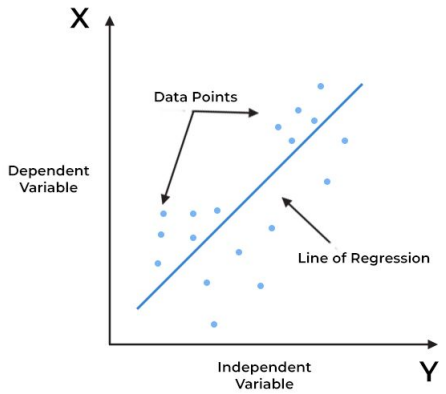

Figure 2.1: Representing the linear relationship between the independent variable and the dependent variable

(b) **Decision Tree Regressor:** Versatile machine learning algorithms that can perform classification, regression, and even multifaceted tasks. They are very powerful algorithms capable of installing complex datasets.

Decision Trees are also the fundamental components of Random Forests, which are among the most powerful Machine Learning algorithms available Today.see figure2.2

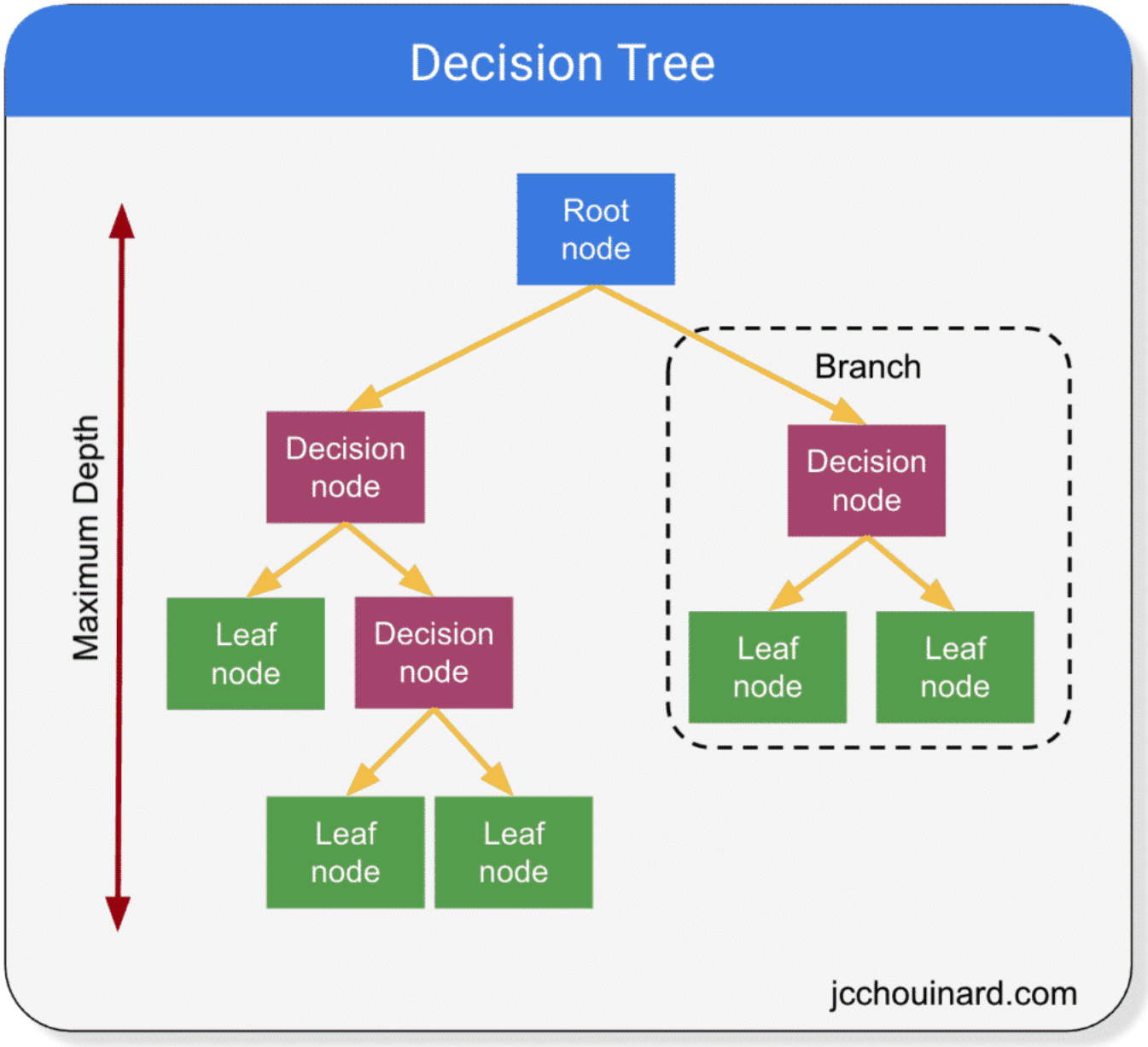

Figure 2.2: A diagram showing how Decision Tree Regressor algorithm works

(c) **Random Forest Regressor:** Algorithm introduces extra randomness when growing trees, instead of searching for the very best feature when splitting a node, it searches for the best feature among a random subset of features. This results in a greater tree diversity, which (once again) trades a higher bias for a lower variance, generally yielding an overall better model.see figure2.3

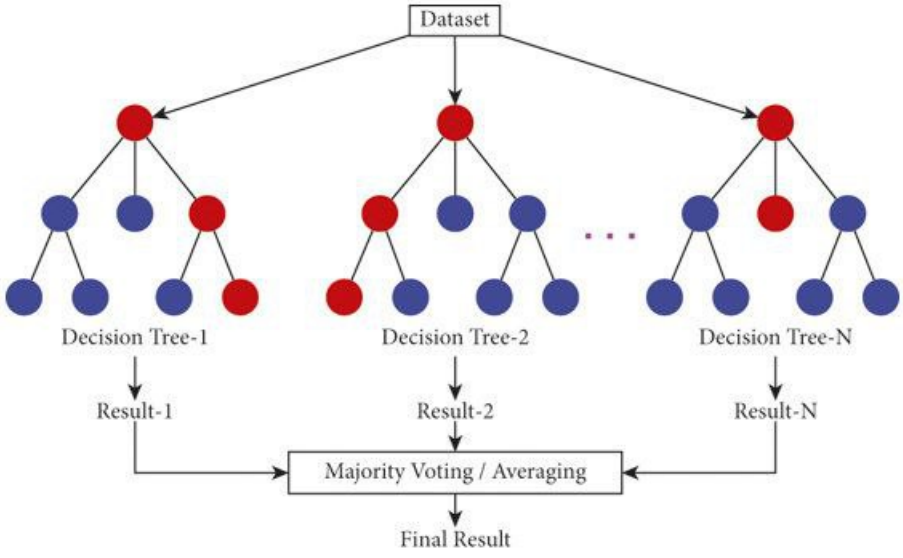

Figure 2.3: A diagram showing how Random Forest Regressor algorithm works

4. **Training the Regression Model**

   In this phase of building a Regression Model, we have all the necessary ingredients to train our model effectively. This involves utilizing our prepared data to teach the model to recognize patterns and make predictions based on the input features.

5. **Evaluating Model Performance**

   Once we have trained our model, it's time to assess its performance. There are various metrics used to evaluate model performance. for Regression tasks, common evaluation metrics are:

   - *Mean Absolute Error (MAE):*MAE is the average of the absolute differ-

ences between predicted and actual values.

$$MAE(X,h) = \frac{1}{m}\sum_{i-1}^{m}|h(x^i) - y^i| \tag{2.1}$$

- *Root Mean Squared Error (RMSE):*A typical performance measure for Regression problems is the Root Mean Square Error (RMSE). It gives an idea of how much error the system typically makes in its predictions.

$$RMSE(X,h) = \sqrt{\frac{1}{m}\sum_{i-1}^{m}(h(x^i) - y^i)^2} \tag{2.2}$$

6. **Deploying the Model and making predictions**

   Deploying the model and making predictions is the final stage in the journey of creating a Regression Model. Once a model has been trained and optimized, it's to integrate it into a production environment where it can provide real-time predictions on new data.

2.3 Conclusion

In this section, we present the main required concepts on Machine Learning and regression models, In the next section we will deal with the collection of the data requires to learn target models.

Chapter 3

Data Set Collection

3.1 Introduction

In the previous section, we have seen the essentials of machine learning. In particular, we detailed regression models. In the current section, we proceed to data collection, as it is the first step in generating the target model. Thus, we follow the next steps:

1. We show the usual standards in collecting data.
2. We present the way we collect data.
3. We describe our data.

3.2 Data Collection Standards

In our project, we should respect the following data collection standards:

- **Quality:** The collected data should be precise and related to subject.
- **Diversity:** The collected data should diverse and be balanced.
- **Quantity:** We should collect enough data to learn target models.
- **Consitancy:** Data must be coherent.

- **Privicy and security:** we avoid including person's names in collected data.
- **Ethical Consideration, transparency and documentation.**

3.3 Data Collection process

In order to define and collect our data, we contacted specialized Doctors and Technicians.

The obtained data in consultation and diagnosis of patients eyes at the hospital (Kouba hospital). Most measurements were taken using the Nidek[1] machine (see figure3.1). Some other data (labels) are values set by specialists (see figure 3.2).The data were organized in an EXCELL shout.[2].

Figure 3.1: Nidek machine

[1]Developed by Nidek, it measures refractive errors in sphere, cylinder, and axis.

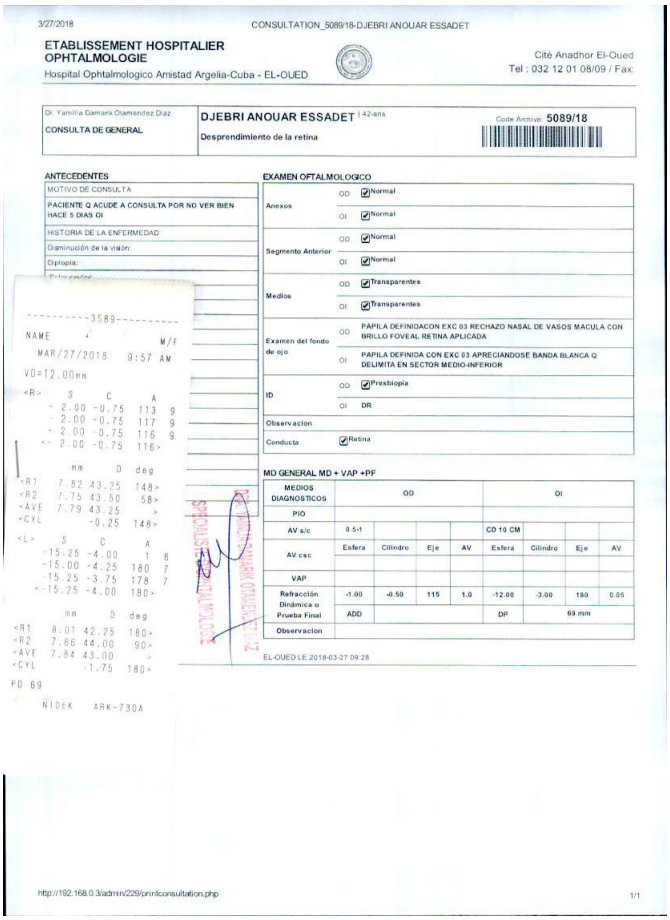

Figure 3.2: Data Sample

3.4 Data describe

Table 3.1 details description of the data we have collected. The first column show the data column name and the second column specifies the data significance.

Table 3.1: Data Description

Data type	**Definition**
VD	Iris Diameter.
Sphere	Is an indicator of the lens strength required to correct nearsightedness, farsightedness, or both.
Cylinder	Indicates the lens power needed to correct astigmatism. Astigmatism is caused by an uneven curvature of the eye's cornea or lens. A minus sign (–) in the prescription corrects nearsighted astigmatism, while a plus sign (+) corrects farsighted astigmatism.
Axis	Is a number between 1 and 180. It indicates the orientation of the astigmatism that appears in the eye.
AV csc (Acuity Vision with the corrected Spectacles)	Measured by putting the right glasses on the person and then evaluating their ability to see things clearly.
AV s/c (Acuity Vision without the Spectacles)	Refers to the measurement of a person's vision acuity without wearing glasses, helping to assess their natural ability to see things clearly.
OI or OS	It means oculus sinister, or the left eye.
OD	It means oculus dexter, or the right eye.
PIO (IOP)	It means intra-ocular pressure.
VAP	If the drop is put to the eye.
AV (Acuity Vision)	Refers to clarity or sharpness of vision. This test tells the doctor if the patient needs glasses and helps find the right corrective lens prescription.
ADD	Represents Addition. It indicates the magnifying power applied to the lower part of suitable lenses to correct age-related farsightedness.
PD	Refers to the distance, in millimetres, between the two pupils of the eyes. It is generally used to centre the lens graduation.

3.5 Conlusion

In this section, we provided a summary of how we obtained the necessary data in collaboration with the specialized medical staff and explaining it while indicating the most important standards for data collection.

Chapter 4

Implementation

4.1 Introduction

Based on the data obtained in the preceding section. In this section, we will deal following the steps:

1. Present the working environment.
2. Describe the process of training regression algorithms, from the stage of loading data to the stage of testing and validation.

4.2 Work environment

To train and experiment with our model, we used the Google Colaboratory environment (Google Colab), with simple CPU. The colab allows us to write and execute python in our browser, with:

- Zero configuration required
- Access to CPU free of charge
- Easy sharing

4.3 Steps of traning the model

4.3.1 Get the data

Firstly, we have mount *Google drive* with *Colab environment*

```
[ ]  from google.colab import drive
     drive.mount('/content/mdrive/')
```

```
Mounted at /content/mdrive/
```

Subsequently, transferring the data to Google Colab and take a look at it:

```
[ ]  import pandas as pd
     The_DATA = pd.read_csv('/content/mdrive/MyDrive/ColabNoteBooks/DataSets/Data.csv', encoding='latin-1')
     The_DATA.head()
```

	Date	Time	VD	RS1	RC1	RA1	REr1	RS2	RC2	RA2	...	RSphere_OD	RCylinder_OD	RAxis_OD	RAV_OD	RSphere_OI
0	03/27/2018	9:57 AM	12	-2.0	-0.75	113	9	-2.00	-0.75	117.0	...	-1	-0.50	115	1.00	-12
1	11/15/2018	7:41 AM	12	-12.5	-1.25	39	9	-12.75	-1.25	45.0	...	-13.75	-1.25	40	0.50	-13.75
2	04/11/2018	8:13 AM	12	0.0	-1.00	5	6	0.00	-1.50	1.0	...	NAT	0.00	180	0.90	NAT
3	04/11/2018	8:16 AM	12	-0.5	-0.50	11	9	-0.50	-0.75	17.0	...	NAT	0.00	180	1.00	NAT
4	07/01/2018	8:20 AM	12	-9.0	-1.75	148	5	-9.50	-2.00	162.0	...	NMCC	0.00	180	0.05	0.75

5 rows × 81 columns

4.3.2 Exploring the data

For more details:

- We use info() method to get more information about the data, in particular the number of rows, and each attribute's type and number of non-null values

```
[ ] data.info()
```

Note that, our data is composed of:

- There are 122 instances.
- Each column is labeled as a type (float, int, or object).
- There are 70 columns that have missing values.

- We also apply the describe () method to provide a summary of numerical features:

```
[ ] Data.describe()
```

	VD	RS1	RC1	RA1	RS2	RC2	RA2	RS3	RC3	RA3	...	VAP_OI	Cylinder_OD
count	122.0	115.000000	122.000000	122.000000	115.000000	122.000000	121.000000	115.000000	122.000000	122.000000	...	122.000000	122.000000
mean	12.0	-2.295652	-1.327869	87.106557	-2.343478	-1.358607	92.231405	-2.278261	-1.303279	88.500000	...	0.073770	-0.229508
std	0.0	4.265784	1.276003	56.975971	4.319664	1.298039	57.931822	4.278384	1.270790	59.559159	...	0.262475	0.842805
min	12.0	-15.750000	-7.000000	2.000000	-15.750000	-6.750000	0.000000	-15.500000	-7.250000	0.000000	...	0.000000	-4.500000
25%	12.0	-3.875000	-1.750000	39.750000	-3.750000	-2.000000	45.000000	-3.875000	-1.750000	34.750000	...	0.000000	0.000000
50%	12.0	-0.750000	-1.000000	82.000000	-1.000000	-1.000000	91.000000	-0.750000	-1.000000	87.500000	...	0.000000	0.000000
75%	12.0	0.000000	-0.500000	140.000000	0.000000	-0.500000	147.000000	0.250000	-0.500000	149.750000	...	0.000000	0.000000
max	12.0	8.000000	0.750000	180.000000	8.000000	1.000000	180.000000	8.000000	0.750000	180.000000	...	1.000000	2.000000

4.3.3 Data preprocessing

Checking correlations with output

In the corresponding code we see the correlation matrix between inputs and the output "RCylinder-OD", by using corr() method in the Pandas library

```
[ ] corr_matrix = Mydata.corr(numeric_only=True)
    corr_matrix['RCylinder_OD'].sort_values(ascending=False)
```

The corr() results show that the column that has the strongest positive correlation with the target is "RC3", so we adopted it as a basis to split the data fairly into train set and test set in the next step.

Split into Train and Test Sets

Firstly, we should create an RC3 category attribute with three categories, by using the cut() method from pandas library (labeled from 1 to 3).

```
[ ] import numpy as np
    Data["RC3_cat"] = pd.cut(Data["RC3"],
                             bins=[-7.50, -3.50,-1.50, np.inf],
                             labels=[1, 2, 3])
    Data["RC3_cat"].value_counts()
```

Then, we split our data into 80% for training, 20% for testing by using the StratifiedShuffleSplit class and checking that the percentage of each category is adequate.

```
[ ] from sklearn.model_selection import StratifiedShuffleSplit
    split = StratifiedShuffleSplit(n_splits=2, test_size=0.2, random_state=42)
    for train_index, test_index in split.split(Data, Data["RC3_cat"]):
      train_set = Data.loc[train_index]
      test_set  = Data.loc[test_index]
```

Data Cleaning

While analyzing the data, we had some issues:

- we came across 9 columns that have missing values. To solve this issue, SimpleImputer is a useful class provided by Scikit-Learn.
- There are 10 columns that contain text values, in this case we used OneHotEncoder class to convert texts to numerical values.
- According to the obtained STD (Standard Deviation), the majority of columns had divergent values, so we used the StandardScaler class to fix this problem.

  In order to address the problems that the data faced there are many data transformation steps that need to be executed in the right order as we've seen. Thus, we use the Pipeline and ColumnTransformer classes, provided by Scikit-Learn, to group all the previous steps in one command.

```
[ ] from sklearn.impute import SimpleImputer
    from sklearn.preprocessing import StandardScaler
    from sklearn.pipeline import Pipeline
    from sklearn.preprocessing import OneHotEncoder
    from sklearn.compose import ColumnTransformer

    numerical_pipeline = Pipeline([
    ('imputer', SimpleImputer(strategy="median")),
    ('std_scaler', StandardScaler()),
    ])

    data_numerical = data.drop(['REr1','REr2','REr3','LEr3','PD','Machine','Annexes OD','Annexes OI',
                                'Anterior Segment OD','Anterior Segment OI','Sphere_OD',
                                'Sphere_OI','AV_OD','AV_OI','AV s/c _OD','AV s/c  _OI'],axis=1)
    numerical_attributes = list(data_numerical)
    Categories_attributes = ['REr1','REr2','REr3','LEr3','PD','Machine','Annexes OD','Annexes OI','Anterior Segment OD',
                                'Anterior Segment OI','Sphere_OD','Sphere_OI','AV_OD','AV_OI','AV s/c _OD','AV s/c  _OI']

    full_pipeline = ColumnTransformer([
    ("num", numerical_pipeline, numerical_attributes),
    ("cat", OneHotEncoder(handle_unknown='ignore'), Categories_attributes),
                                    ])

    data_prepared = full_pipeline.fit_transform(data)
```

4.3.4 Training Models

After processing the dataset and solving all related problems, now the dataset is ready for model training. We experiment our data through various algorithms, then we check their predictions:

Linear Regression

```
[ ] from sklearn.linear_model import LinearRegression
    Linear_Regression = LinearRegression()
    Linear_Regression.fit(data_prepared, data_label)
    some_data = data.iloc[:5]
    some_label = data_label.iloc[:5]
    some_data_prepared = full_pipeline.transform(some_data)
    print("Predictions:", Linear_Regression.predict(some_data_prepared))
    some_label.head()

    Predictions: [-2.49997108e+00  2.96829948e-05 -1.00002168e+00 -7.49955479e-01
     -5.00025100e-01]
    48   -2.50
    27    0.00
    66   -1.00
    99   -0.75
    47   -0.50
    Name: RCylinder_OD, dtype: float64
```

We have done the same experiments on Decision Tree and Random Forest Algorithms (see the detailed codes in Appendix 1).

4.3.5 Evaluation of Models

To evaluate and check the performance of each algorithm, we use the *Root Mean Square Error*(RSME) metrics.

- Linear Regression

```
5.9297694304743125e-05
```

- Decision Tree Regression

```
0.0
```

- Random Forest Regression

```
0.349995066239158
```

As we can note here, the *Root Mean Square Error* of the Decision Tree Regression is "0.0". In fact, this is a fake value, since when checking it using the cross-validation techniques it gives a greater value compared to other algorithms.

```
[ ] from sklearn.model_selection import cross_val_score
    scores                   = cross_val_score(Decision_TreeRegressor, data_prepared, data_label, scoring="neg_mean_squared_error", cv=5)
    Decision_Tree_rmse_scores = np.sqrt(-scores.mean())
    print("Root Mean Squared Error for meam of scores :\n", Decision_Tree_rmse_scores)
    Decision_Tree_RMSE = Decision_Tree_rmse_scores

Root Mean Squared Error for meam of scores :
 1.2059577979872744
```

Finally, as a result, we can say that the algorithm with the best evaluation is Random Forest Regressor.

4.3.6 Testing the best model

After evaluating and selecting the best model, we used the predict () method to test this model on unseen data processed using the full pipeline.

```
[ ]  some_data_test          = datatest.iloc[10:15]
     some_data_test_label    = datatest_label.iloc[10:15]
     some_data_test_prepared = full_pipeline.transform(some_data_test)
     print("Predictions:", RandomForest_Regressor.predict(some_data_test_prepared))
     some_data_test_label.head()
```

```
Predictions: [-1.955 -1.601 -2.08  -0.035 -0.105]
39     0.00
64    -2.75
102   -3.00
35     0.50
111   -0.50
Name: RCylinder_OD, dtype: float64
```

4.3.7 Conclusion

In this chapter, we presented details about the implementation of our work. We have used Google Collaboratory with Python 3 as environment. We have followed a systematic way to prepare, learn and test targeted models. Through this study, we conclude that the Random Forest model was the best. We tested it and checked it on unseen data. It gives mostly best predictions.

conclusion

By systematically preparing data, training the model, and evaluating it, the project successfully identified random forest regression as the most effective algorithm for predicting eye measurements. Applying this model can greatly improve the accuracy of eyeglass prescriptions, ultimately enhancing patient outcomes. Integrating machine learning into medical diagnostics showcases the potential of AI to revolutionize healthcare practices by providing precise and reliable solutions. Overall, this project not only demonstrates the use of advanced machine learning techniques in practical healthcare settings but also highlights the importance of comprehensive data processing and model evaluation in achieving reliable results. The project aims to collect more data to train the model further, which will improve its accuracy and effectiveness in various situations. We plan to develop a desktop application that includes the selected model, allowing specialists to input the necessary data and get predictions directly from the model

Bibliography

[1] Propose a title here. http://www.geeksforgeeks.ogs/regression-in-machine-learning. Accessed: 2024-05-30.

[2] Propose a title here. https://www.healthline.com/health/how-to-read-eye-prescriptioneye-checkups. Accessed: 2024-05-30.

[3] Propose a title here. `https://www.warbyparker.com/learn/how-to-read-eye-prescription`. Accessed: 2024-05-30.

[4] Jason Brownlee. *Machine learning mastery with Python: understand your data, create accurate models, and work projects end-to-end*. Machine Learning Mastery, 2016.

[5] Aurélien Géron. *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow*. " O'Reilly Media, Inc.", 2022.

[6] Dipanjan Sarkar, Raghav Bali, and Tushar Sharma. Practical machine learning with python. *Book" Practical Machine Learning with Python*, pages 25–30, 2018.

[7] Shai Shalev-Shwartz and Shai Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014.

[8] K Sudhaman, Mahesh Akuthota, and Sandip Kumar Chaurasiya. A review on the different regression analysis in supervised learning. *Bayesian Reasoning and Gaussian Processes for Machine Learning Applications*, pages 15–32, 2022.

[9] Manohar Swamynathan. *Mastering machine learning with python in six steps: A practical implementation guide to predictive data analytics using python*. Springer, 2017.

[10] Lisa Tagliaferri, Michelle Morales, Ellie Birbeck, and Alvin Wan. Machine learning projects: Python. *New york: Digital Ocean*, 2019.

[11] Sandra Vieira, Rafael Garcia-Dias, and Walter Hugo Lopez Pinaya. A step-by-step tutorial on how to build a machine learning model. In *Machine learning*, pages 343–370. Elsevier, 2020.

Appendix A

Note Book: Code Python

```
from google.colab import drive
drive.mount('/content/mdrive/')

import pandas as pd
The_DATA = pd.read_csv('/content/mdrive/MyDrive/ColabNoteBooks/DataSets/Data.csv', encoding='latin-1')
The_DATA.head()

copy_data = The_DATA
Data      = copy_data.drop(['Date', 'Time', 'RAxis_OD', 'RAV_OD', 'RSphere_OD',
 'RSphere_OI', 'RCylinder_OI', 'RAxis_OI', 'RAV_OI', 'ADD', 'DP'], axis=1)

Data.info()

Data.describe()

corr_matrix = Data.corr(numeric_only=True)
corr_matrix['RCylinder_OD'].sort_values(ascending=False)

import numpy as np
Data["RC3_cat"] = pd.cut(Data["RC3"],
                         bins=[-7.50, -3.50,-1.50, np.inf],
                         labels=[1, 2, 3])
Data["RC3_cat"].value_counts()

"""# **Spliting the data**"""

from sklearn.model_selection import StratifiedShuffleSplit
split = StratifiedShuffleSplit(n_splits=2, test_size=0.2, random_state=42)
for train_index, test_index in split.split(Data, Data["RC3_cat"]):
  train_set = Data.loc[train_index]
```

```
  test_set  = Data.loc[test_index]

train_set.info()

test_set.info()

train_set["RC3_cat"].value_counts()/len(train_set)

test_set["RC3_cat"].value_counts()/len(test_set)

for set_ in (train_set, test_set):
    set_.drop("RC3_cat", axis=1, inplace=True)

data        = train_set.drop("RCylinder_OD", axis=1)
data_label = train_set["RCylinder_OD"].copy()
train_copy = train_set.copy()

train_set.plot(kind="scatter", x="RC3", y="RCylinder_OD")

"""# **Fix features problems**"""

from sklearn.impute import SimpleImputer
from sklearn.preprocessing import StandardScaler
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import OneHotEncoder
from sklearn.compose import ColumnTransformer

numerical_pipeline = Pipeline([
('imputer', SimpleImputer(strategy="median")),
('std_scaler', StandardScaler()),
])

data_numerical = data.drop(['REr1','REr2','REr3','
LEr3','PD','Machine','Annexes OD','Annexes OI',
                            'Anterior Segment OD','Anterior Segment OI','Sphere_OD',
                            'Sphere_OI','AV_OD','AV_OI','AV s/c _OD','AV s/c  _OI'],axis=1)
numerical_attributes = list(data_numerical)
Categories_attributes = ['REr1','REr2','REr3',
 'LEr3','PD','Machine','Annexes OD','Annexes OI','Anterior Segment OD',
                            'Anterior Segment OI','Sphere_OD','Sphere_OI',
                            'AV_OD','AV_OI','AV s/c _OD','AV s/c  _OI']

full_pipeline = ColumnTransformer([
("num", numerical_pipeline, numerical_attributes),
("cat", OneHotEncoder(handle_unknown='ignore'), Categories_attributes),
                                   ])
```

```
data_prepared = full_pipeline.fit_transform(data)

"""# **Training models and make prediction**

# **1_ LinearRegression**
"""

from sklearn.linear_model import LinearRegression
Linear_Regression = LinearRegression()
Linear_Regression.fit(data_prepared, data_label)
some_data = data.iloc[:5]
some_label = data_label.iloc[:5]
some_data_prepared = full_pipeline.transform(some_data)
print("Predictions:", Linear_Regression.predict(some_data_prepared))
some_label.head()

"""# **2_ DecisionTreeRegressor**"""

from sklearn.tree import DecisionTreeRegressor
Decision_TreeRegressor = DecisionTreeRegressor()
Decision_TreeRegressor.fit(data_prepared, data_label)
print("Predictions:", Decision_TreeRegressor.predict(some_data_prepared))
some_label.head()

"""# **3_ RandomForestRegressor**"""

from sklearn.ensemble import RandomForestRegressor
RandomForest_Regressor = RandomForestRegressor()
RandomForest_Regressor.fit(data_prepared, data_label)
print("Predictions:", RandomForest_Regressor.predict(some_data_prepared))
some_label.head()

"""# **RMSE measurement for the three models**

# ***1 _ linearReression***
"""

from sklearn.metrics import mean_squared_error
data_predictions = Linear_Regression.predict(data_prepared)
linear_mse= mean_squared_error(data_label, data_predictions)
linear_rmse= np.sqrt(linear_mse)
linear_rmse

"""# ***2 _ DecisionTreeRegressor***"""
```

```
data_predictions_2 = Decision_TreeRegressor.predict(data_prepared)
Decision_Tree_mse  = mean_squared_error(data_label, data_predictions_2)
Decision_Tree_rmse = np.sqrt(Decision_Tree_mse)
Decision_Tree_rmse

"""# ***3 - RandomForestRegressor***"""

data_predictions_3 = RandomForest_Regressor.predict(data_prepared)
RandomForest_mse   = mean_squared_error(data_label, data_predictions_3)
RandomForest_rmse  = np.sqrt(RandomForest_mse)
RandomForest_rmse

"""# **Evaluation the DecisionTreeRegressor with cross validation**"""

from sklearn.model_selection import cross_val_score
scores  = cross_val_score(Decision_TreeRegressor,
data_prepared, data_label, scoring="neg_mean_squared_error", cv=5)
Decision_Tree_rmse_scores = np.sqrt(-scores.mean())
print("Root Mean Squared Error for meam of scores :\n", Decision_Tree_rmse_scores)
Decision_Tree_RMSE = Decision_Tree_rmse_scores

"""# **Testing the model**"""

datatest = test_set.drop("RCylinder_OD", axis=1)
datatest_label = test_set["RCylinder_OD"].copy()

some_data_test          = datatest.iloc[10:15]
some_data_test_label    = datatest_label.iloc[10:15]
some_data_test_prepared = full_pipeline.transform(some_data_test)
print("Predictions:", RandomForest_Regressor.predict(some_data_test_prepared))
some_data_test_label.head()
```