Machine Learning Engineer Nanodegree Capstone Report

Andreas Quauke

September 9th, 2020

1. Definition

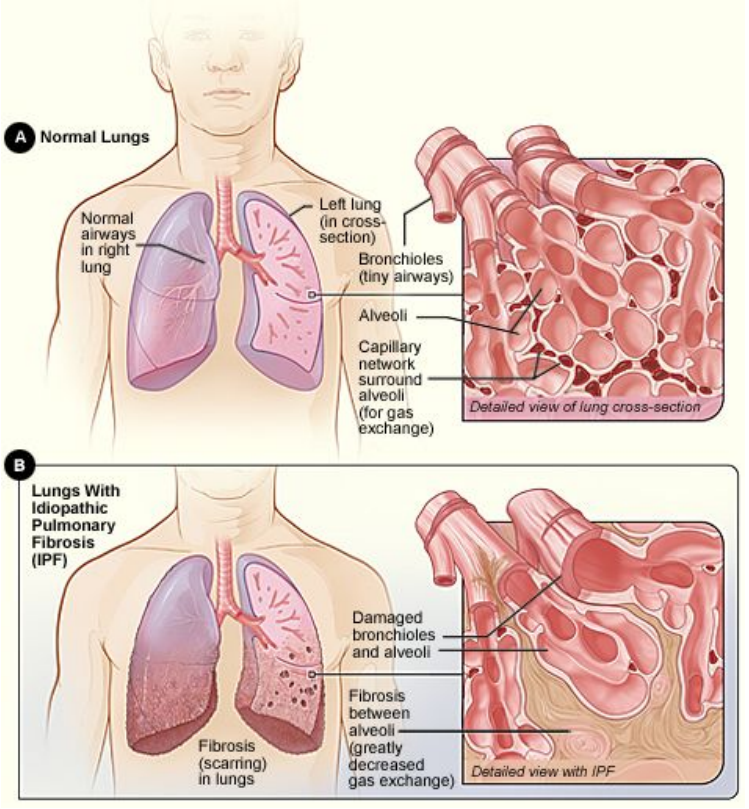
For my capstone project I participated in the *OSIC Pulmonary Fibrosis Progression* Challenge on Kaggle. The following report summarizes the results of my work.

1.1. Project Overview

To give a comprehensive overview I will first briefly explain the problem domain and the medical background and secondly describe what kind of data was available to work with.

1.1.1. Problem Domain and Background

The topic of my capstone project is based on a recent Kaggle competition that deals with a lung disease called *pulmonary fibrosis*. "Pulmonary fibrosis is a condition in which the lungs become scarred over time. Symptoms include shortness of breath, a dry cough, feeling tired, weight loss, and nail clubbing. Complications may include pulmonary hypertension, respiratory failure, pneumothorax, and lung cancer."[1]



Pulmonary fibrosis is a disorder with no known cause

[1] Wikipedia contributors. "Pulmonary Fibrosis." *Wikipedia*, 12 Aug. 2020, en.wikipedia.org/wiki/Pulmonary_fibrosis.

and no known cure, created by scarring of the lungs (s. figure above[2]). People who get this disease would want to know their prognosis. And that is where a troubling disease becomes frightening for the patient: "Outcomes can range from long-term stability to rapid deterioration, but doctors aren't easily able to tell where an individual may fall on that spectrum. Current methods make fibrotic lung diseases difficult to treat, even with access to a chest CT scan. In addition, the wide range of varied prognoses create issues organizing clinical trials. Finally, patients suffer extreme anxiety—in addition to fibrosis-related symptoms—from the disease's opaque path of progression."[3]

In this capstone project - like in the respective competition on Kaggle - I would like to predict a patient's severity of decline in lung function based on a CT scan of their lungs and determine lung function based on output from a spirometer, which measures the volume of air inhaled and exhaled. The goal is to predict the patient's *Forced Vital Capacity* (FVC) by analysing her CT image and her prevailing FVC measurements. The FVC measurement shows the amount of air a person can forcefully and quickly exhale after taking a deep breath.[4]

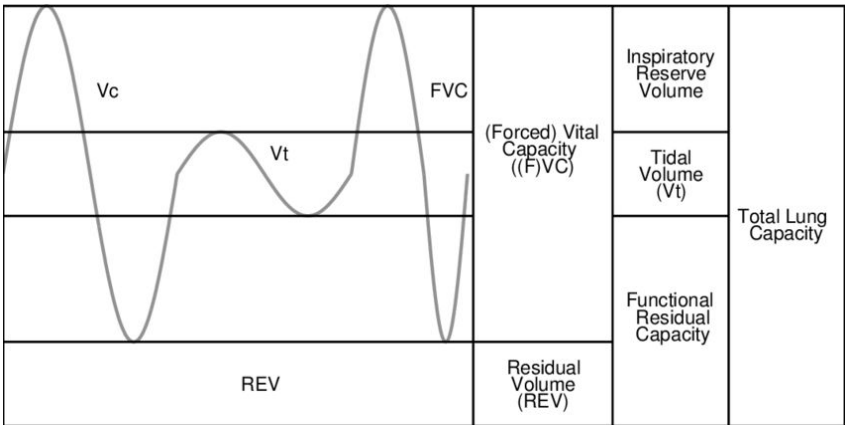


Figure 2 shows the relations between different lung metrics schematically[5]. The total capacity of a lung is the sum of the FVC, i.e. the maximum of volume a human can inhale and exhale, and the residual volume (REV) that cannot be exhaled.

---

[2] Mirchandani, Jagdmir. "Insightful EDA on Meta Data & Dicom Files." Kaggle, 1 Sept. 2020, www.kaggle.com/jagdmir/insightful-eda-on-meta-data-dicom-files.

[3] OSIC Pulmonary Fibrosis Progression. (n.d.). Kaggle. Retrieved August 18, 2020, from https://www.kaggle.com/c/osic-pulmonary-fibrosis-progression

[4] FEV1 and FVC: What Do They Mean for You? (2018, February 11). Lung Health Institute. https://lunginstitute.com/blog/fev1-and-fvc/

[5] Mccaughey, Euan. (2014). Abdominal Functional Electrical Stimulation to improve respiratory function in acute and sub-acute tetraplegia.

My personal motivation is to help patients and their families better understand their prognosis when they are first diagnosed with this disease. Indeed, I have never worked in the medical field before, but this is exactly why I find it interesting and challenging to understand how machine learning techniques can help to make people's life better.

1.2. Problem Statement

The problem to solve is to predict a patient's FVC in weekly steps over a given period of time in the future. By assessing the patient's lung CT and her given FVC measurements of her past examinations, I get indicators if the disease is progressing, stabilizing or even getting better. The CT images can be classified in terms of progression of the disease, and the FVC history also gives indications of its recent development.
The FVC values are predicted in milliliters (ml) and therefore are clearly quantifiable and the quality of the predictions are measurable as well. As there are many patients affected by this disease, the problem is also replicable.

The proposed solution to this problem consists of two steps:
Firstly, I apply an image processing algorithm that gives me an indication what the current status of the disease is, e.g. by measuring the degree of honeycombing[6] on the lung tissue or the lung volume. I can then add these newly discovered features to the existing dataset.
Secondly, I build a forecasting algorithm like Gradient Boosting to predict the time series for future FVC values based on the existing and newly added features.

1.3. Metrics

This capstone project is evaluated on a modified version of the *Laplace Log Likelihood*. "In medical applications, it is useful to evaluate a model's confidence in its decisions. Accordingly, the metric is designed to reflect both the accuracy and certainty of each prediction. For each true FVC measurement, you will predict both an FVC and a confidence measure (standard deviation σ). The metric is computed as:

$$\sigma_{clipped} = max(\sigma, 70),$$

$$\Delta = min(|FVC_{true} - FVC_{predicted}|, 1000),$$

$$metric = -\frac{\sqrt{2}\Delta}{\sigma_{clipped}} - \ln(\sqrt{2}\sigma_{clipped})$$

[6] Wikipedia contributors. (2017, November 4). Honeycombing. Wikipedia. https://en.wikipedia.org/wiki/Honeycombing

The error is thresholded at 1000 ml to avoid large errors adversely penalizing results, while the confidence values are clipped at 70 ml to reflect the approximate measurement uncertainty in FVC. The final score is calculated by averaging the metric across all `Patient_Weeks` (three per patient) of the test set. Note that metric values will be negative and higher is better."[7]

2. Analysis

2.1. Data Exploration

2.1.1. Datasets and Inputs

For this project I will use the dataset provided by Kaggle for their competition. It is publicly accessible on their website, where the structure of the data is also described:[8]

"In the dataset, you are provided with a baseline chest CT scan and associated clinical information for a set of patients. A patient has an image acquired at time Week = 0 and has numerous follow up visits over the course of approximately 1-2 years, at which time their FVC is measured.

- In the training set, you are provided with an anonymized, baseline CT scan and the entire history of FVC measurements.
- In the test set, you are provided with a baseline CT scan and only the initial FVC measurement. You are asked to predict the final three FVC measurements for each patient, as well as a confidence value in your prediction.

Since this is real medical data, you will notice the relative timing of FVC measurements varies widely. The timing of the initial measurement relative to the CT scan and the duration to the forecasted time points may be different for each patient."

As input data a tabular dataset with all patient records is given. The following table is an example of the first five records of one single patient.

[7] Dataset OSIC Pulmonary Fibrosis Progression. (n.d.). Kaggle. Retrieved August 18, 2020, from https://www.kaggle.com/c/osic-pulmonary-fibrosis-progression/data

[8] Dataset OSIC Pulmonary Fibrosis Progression. (n.d.). Kaggle. Retrieved August 18, 2020, from https://www.kaggle.com/c/osic-pulmonary-fibrosis-progression/data

	Patient	Weeks	FVC	Percent	Age	Sex	SmokingStatus
0	ID00007637202177411956430	-4	2315	58.253649	79	Male	Ex-smoker
1	ID00007637202177411956430	5	2214	55.712129	79	Male	Ex-smoker
2	ID00007637202177411956430	7	2061	51.862104	79	Male	Ex-smoker
3	ID00007637202177411956430	9	2144	53.950679	79	Male	Ex-smoker
4	ID00007637202177411956430	11	2069	52.063412	79	Male	Ex-smoker

In this table of the test set there are 1549 entries of 176 unique patients.

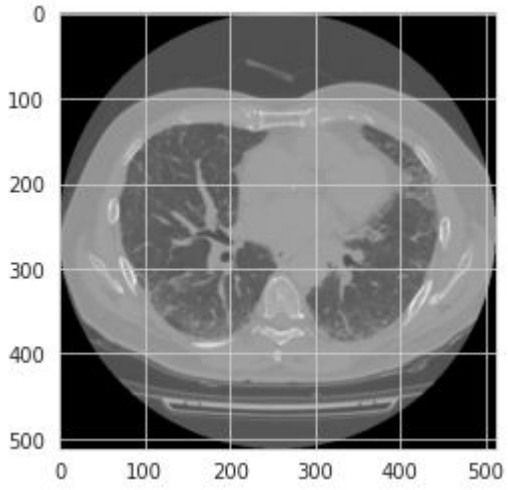
2.1.2. Description of the features

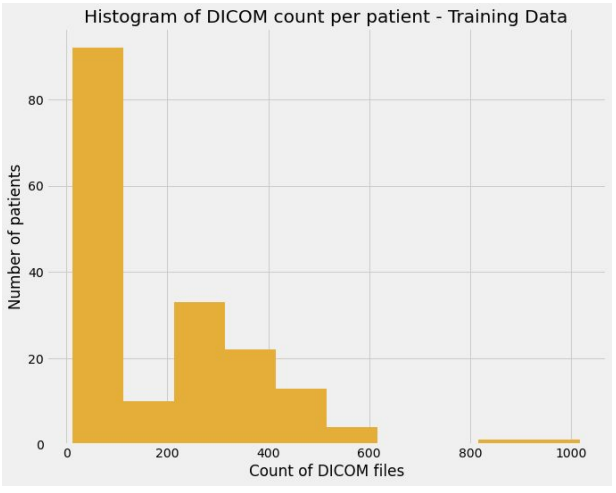
The tabular data contains the following features:

- `Patient`: a unique Id for each patient
- `Weeks`: the relative number of weeks pre/post the baseline CT
- `FVC`: the recorded lung capacity in ml
- `Percent`: a computed field which approximates the patient's FVC as a percent of the typical FVC for a person of similar characteristics
- `Age`: the patient's age in years
- `Sex`: the patient sex as string
- `SmokingStatus`: the patients smoking status, which can be "Ex-Smoker", "currently smokes" or "never smoked"

2.1.3. CT Scans

Besides the tabular data, there is one CT scan given for each single patient. This CT is always recorded in Week 0. This gives an indication of how many weeks there are in between the CT scan and the FVC measurements.





Each CT scan consists of multiple slices (s. figure on the left) that show a horizontal cut through the patient's torso. The number of slices varies from patient to patient (s. figure on the right) and can be in between 16 and over 1000 slices per scan. Moreover, each scan picture (i.e., each slice) contains the scanner's metadata which again consists of data like the distance between pixels - horizontally as well as vertically - the thickness of the slices or the position of the slice within the patient.

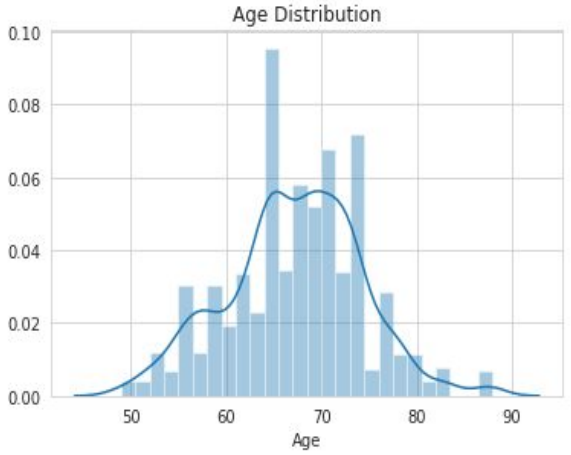
This is an example for the meta data of one picture:

```
Dataset.file_meta -------------------------------
(0002, 0000) File Meta Information Group Length  UL: 200
(0002, 0001) File Meta Information Version       OB: b'\x00\x01'
(0002, 0002) Media Storage SOP Class UID         UI: CT Image Storage
(0002, 0003) Media Storage SOP Instance UID      UI: 2.25.529504783016724814091859207492743889 98
(0002, 0010) Transfer Syntax UID                 UI: Explicit VR Little Endian
(0002, 0012) Implementation Class UID            UI: 1.2.276.0.7230010.3.0.3.6.1
(0002, 0013) Implementation Version Name         SH: 'OSIRIX_361'
(0002, 0016) Source Application Entity Title     AE: 'ANONYMOUS'
-------------------------------------------------
(0008, 0008) Image Type                          CS: ['ORIGINAL', 'PRIMARY', 'AXIAL', 'HELIX']
(0008, 0018) SOP Instance UID                    UI: 2.25.529504783016724814091859207492743889 98
(0008, 0060) Modality                            CS: 'CT'
(0008, 0070) Manufacturer                        LO: 'Philips'
(0008, 1090) Manufacturer's Model Name           LO: 'Brilliance 40'
(0010, 0010) Patient's Name                      PN: 'ID00368637202296470751086'
(0010, 0020) Patient ID                          LO: 'ID00368637202296470751086'
(0010, 0040) Patient's Sex                       CS: ''
(0012, 0063) De-identification Method            LO: 'Table;'
(0018, 0015) Body Part Examined                  CS: 'Chest'
(0018, 0050) Slice Thickness                     DS: "2.0"
(0018, 0060) KVP                                 DS: "120.0"
(0018, 0088) Spacing Between Slices              DS: "-1.0"
(0018, 1120) Gantry/Detector Tilt                DS: "0.0"
(0018, 1130) Table Height                        DS: "113.0"
(0018, 1140) Rotation Direction                  CS: 'CW'
(0018, 1151) X-Ray Tube Current                  IS: "214"
(0018, 1210) Convolution Kernel                  SH: 'C'
(0018, 5100) Patient Position                    CS: 'HFS'
(0020, 000d) Study Instance UID                  UI: 2.25.785568186214358425841837026883137 6965
(0020, 000e) Series Instance UID                 UI: 2.25.706935369208650862162855362356502 64865
(0020, 0010) Study ID                            SH: ''
(0020, 0013) Instance Number                     IS: "270"
(0020, 0032) Image Position (Patient)            DS: [-220, -62, 57.4000244]
(0020, 0037) Image Orientation (Patient)         DS: [1, 0, 0, 0, 1, 0]
(0020, 0052) Frame of Reference UID              UI: 2.25.108537503034797627859324176733856619672
(0020, 1040) Position Reference Indicator        LO: ''
(0020, 1041) Slice Location                      DS: "57.4"
(0028, 0002) Samples per Pixel                   US: 1
(0028, 0004) Photometric Interpretation          CS: 'MONOCHROME2'
(0028, 0010) Rows                                US: 512
(0028, 0011) Columns                             US: 512
(0028, 0030) Pixel Spacing                       DS: [0.796875, 0.796875]
(0028, 0100) Bits Allocated                      US: 16
(0028, 0101) Bits Stored                         US: 12
(0028, 0102) High Bit                            US: 11
```

```
(0028, 0103) Pixel Representation                US: 0
(0028, 1050) Window Center                       DS: "-500.0"
(0028, 1051) Window Width                        DS: "-1500.0"
(0028, 1052) Rescale Intercept                   DS: "-1024.0"
(0028, 1053) Rescale Slope                       DS: "1.0"
(7fe0, 0010) Pixel Data                          OW: Array of 524288 elements
```

2.2. Exploratory Visualization

In the following section I would like to give a brief overview of how the data is distributed.



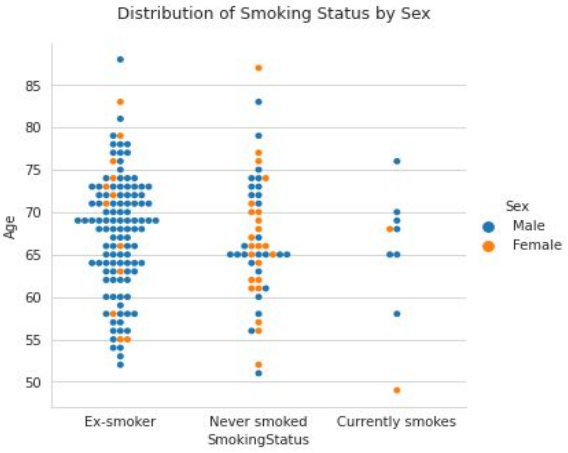
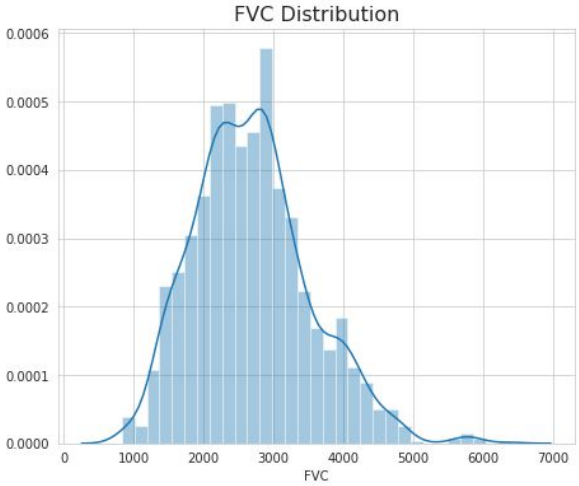
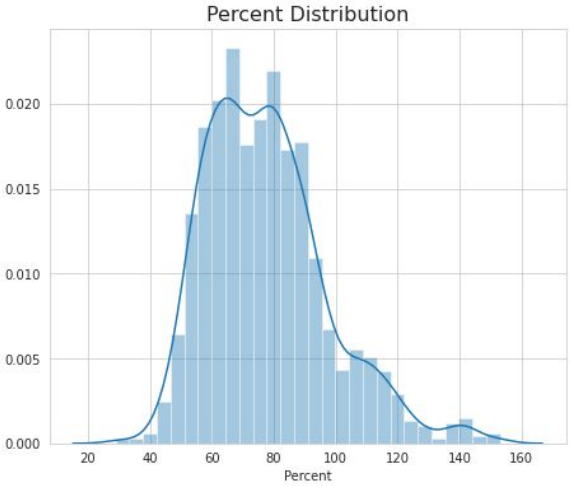


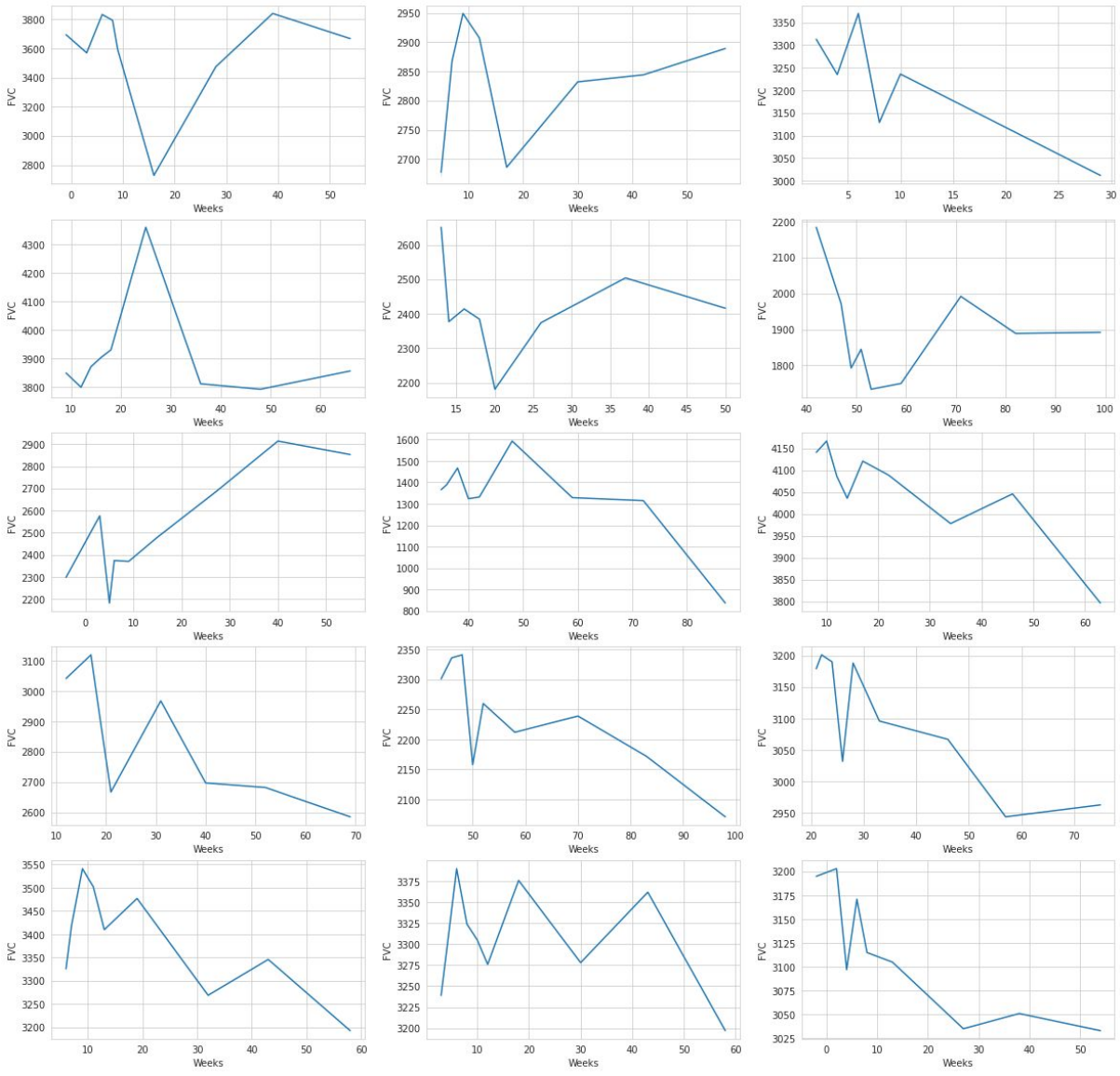
Figure “Age Distribution” shows that most of the patients are elderly people, mostly between 60 and 80 years.
Figure “Distribution of Smoking Status by Sex” shows that most of the patients are male, which may be a possible bias when predictions are made based on this dataset. It is also noteworthy that the majority of the patients have been ex-smokers or currently smoke, which could be an indication that this disease is reinforced by smoking. Nevertheless, most female patients have never smoked in their life.





Looking at the distributions of the FVC and the FVC percentage of a “comparable” healthy patient, it seems that both are almost normally distributed with a mean FVC around 2700 and a mean Percentage of 77%.

In the charts below, 15 different patients with their individual FVC measurements are shown.



2.3. Algorithms and Techniques

For this report I use two kinds of algorithms:

1. An algorithm that extracts image data from the CT scans and their meta data and converts it into new features. This algorithm consists mainly of simple algebraic matrix operations performed in numpy.

2. Another algorithm to predict the gradient coefficient of every single patient's FVC curve. For this task we use scikit-learn's Gradient Boosting algorithm as it performs very reliably on regression problems like this and it is also very robust against overfitting.

I will explain the detailed implementation in 3.2.

2.4. Benchmark

For this particular problem, there is no available public benchmark. However, I will define an evaluation metric, a modified version of Laplace Log Likelihood (cf. "Metrics" in 1.3 above).

The minimum benchmark for my model to beat is to provide a better prediction than you would get by predicting the mean and standard deviation for each patient.

"-8.023 is the default score to beat while cross-validating models on train data. Any model scoring worse than this is not useful. You can get this default score to beat for each (fold of) validation data as well."[9]

The best possible outcome on this metric would be $-\ln(\sqrt{2}\sigma_{clipped})$ and as $sigma_{clipped}$ has a defined minimum of 70, this yields ~ -4.595.

3. Methodology

In this chapter I will describe what methodology I have used to preprocess the data and to implement the algorithms mentioned above.

3.1. Data Preprocessing

3.1.1. Tabular Data

For the tabular data I have performed the following steps of basic processing:

1. I had to check for NaN values and delete the respective rows in the dataset (which was not the case in the given dataset)
2. As the prediction task is to predict a FVC value for a given patient in a given week, I had to create a modified dataset where all the data of the *unique patients* are summarized. Additional metrics for each patient like first and last

[9] Vopani. (2020, July 17). OSIC: Understanding Laplace Log Likelihood. Kaggle. https://www.kaggle.com/rohanrao/osic-understanding-laplace-log-likelihood

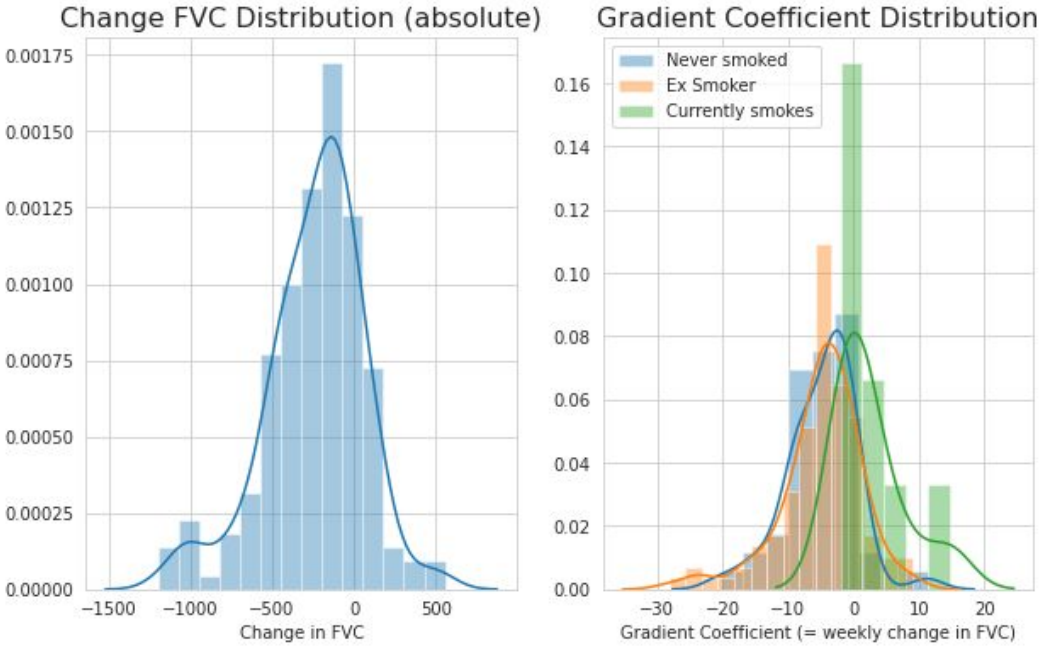
FVC value measured (“FVC_First” and “FVC_Last”), maximum and minimum FVC value (“Max_FVC” and “Min_FVC”) as well as spread (or difference) between first and last measurement (“Diff_FVC_Abs”) and standard deviation of the given measurements of each patient (“Std_FVC”) were calculated.

3. On the given measurements of each patient (around 10 measurements per patient over the time) I have performed a simple linear regression that shows an approximation of the patient’s FVC decay (or recovery) and I have stored the gradient coefficient of this linear regression for each patient (“Gradient_Coef”).
4. I converted the categorial values of “Sex” and “SmokingStatus” into boolean values.
5. As a result I got a dataframe of all patients that looks as follows:

	FVC_First	FVC_Last	Diff_FVC_Abs	Gradient_Coef	Max_FVC	Min_FVC	Std_FVC	Percent	Age	Sex	SmokingStatus
ID00007637202177411956430	2315	2057	-258	-3.167126	2315	2000	96.856136	58.253649	79	Male	Ex-smoker
ID00009637202177434476278	3660	3214	-446	-9.379955	3895	3214	197.367677	85.282878	69	Male	Ex-smoker
ID00010637202177584971671	3523	2518	-1005	-17.042803	3523	2474	358.433901	94.724672	60	Male	Ex-smoker
ID00011637202177653955184	3326	3193	-133	-4.548925	3541	3193	113.773899	85.987590	72	Male	Ex-smoker
ID00012637202177665765362	3418	2971	-447	-8.543079	3759	2971	212.432211	93.726006	65	Male	Never smoked

Male	Female	NeverSmoked	ExSmoker	CurrentlySmokes
1	0	0	1	0
1	0	0	1	0
1	0	0	1	0
1	0	0	1	0
1	0	1	0	0

A closer look at this newly generated feature dataframe shows that also the calculated values of gradient coefficient and spread are nearly normally distributed.



3.1.2. CT Image Data

Preprocessing of the CT images takes more effort than tabular data. The challenge here is to get from a original CT image (cf. figure "CT image" in 2.1.3) to the features that have to be extracted. For this analysis I would like to extract the patient's lung volume in cm² and the percentage of fibrotic tissue in the patient lungs.

The basic principle is to translate a CT scanner's images, measured in Hounsfield units[10] to pixels. "Normal lung attenuation is between -600 and -700 Hounsfield units" [11] and this is what the filtering algorithm below is based on:

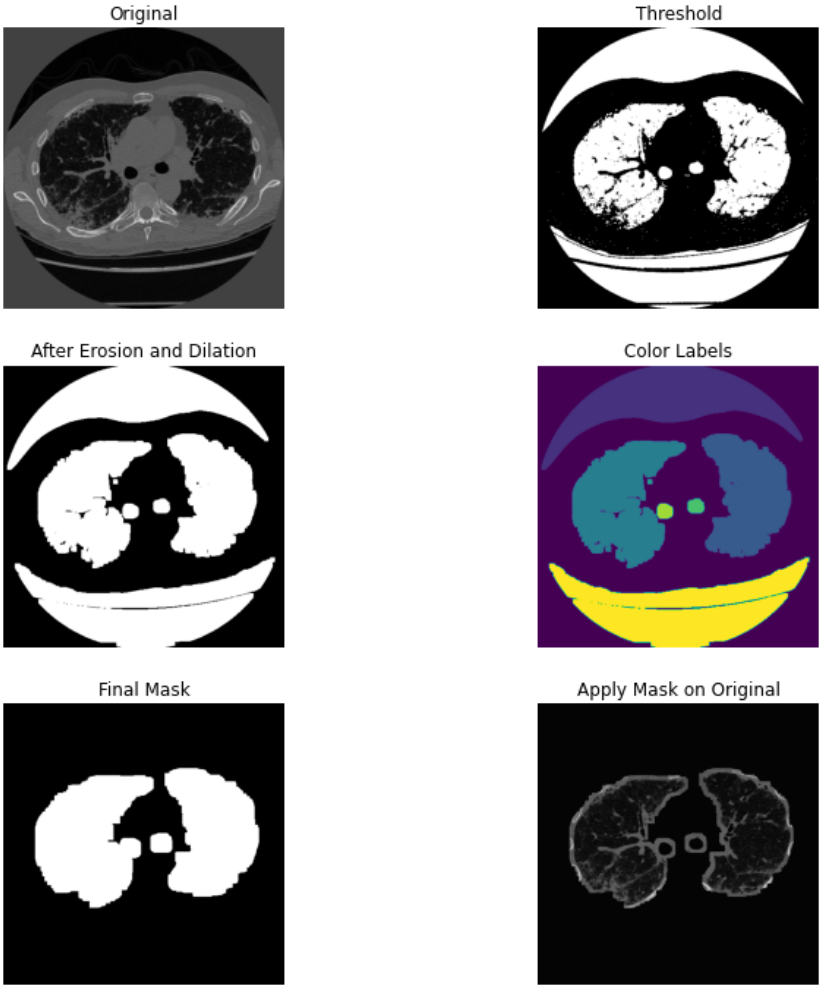


Figure 3.1.2 "Lung Mask"

[10] Wikipedia contributors. "Hounsfield Scale." Wikipedia, 13 Aug. 2020, en.wikipedia.org/wiki/Hounsfield_scale.

[11] Kazerooni, Ella, and Barry Gross. Cardiopulmonary Imaging. Lippincott Williams & Wilkins, 2004.

The procedure is described very detailed by Howard Chen in his article about DICOM Processing and Segmentation in Python:[12]

1. First, I load the images in dicom format[13] for each patient using Python's pydicom library[14].
2. I convert all images to arrays of pixels with pythons pixel_array function (s. fig. 3.1.2 "Original")
3. I find the average pixel value near the lungs to renormalize washed out images and after that I use scikit-learn's Kmeans with two target clusters to separate foreground (soft tissue / bone) and background (lung/air) (s. fig. 3.1.2 "Threshold")
4. Next, I erode away the finer elements, then I dilate to include some of the pixels surrounding the lung (s. fig. 3.1.2 "After Erosion and Dilation")
5. In the following step, different labels are displayed in different colors (s. fig. 3.1.2 "Color Labels") using scikit-image's morphology module[15]
6. I filter the labeled areas by size and proportions and after just the lungs are left, I do another large dilation in order to fill in and out the lung mask (s. fig. 3.1.2 "Final Mask")
7. Finally I use plain numpy to multiply the final mask matrix by the original image pixel matrix and I get only the lungs in the original image (s. fig. 3.1.2 "Final Mask")

3.2. Implementation

The implementation of the algorithms are completely done in Python and I used a jupyter notebook as this was also the requirement for the Kaggle competition.
In the following sections I describe the Basic Workflow of my project as well as the technology and libraries I used.

3.2.1. Basic Workflow

The workflow of this capstone project was designed as follows:

[12] Chen, Howard. "DICOM Processing and Segmentation in Python." Radiology Data Quest, Radiology Data Quest, 6 Jan. 2019, www.raddq.com/dicom-processing-segmentation-visualization-in-python.

[13] "DICOM - Digital Imaging and Communications in Medicine." DICOM, dicomstandard.org. Accessed 5 Sept. 2020.

[14] Pydicom. "Pydicom/Pydicom." GitHub, github.com/pydicom/pydicom. Accessed 5 Sept. 2020.

[15] https://scikit-image.org/docs/dev/api/skimage.morphology.html. Accessed 5 Sept. 2020.

1. Clean and preprocess the data (cf. 3.1. “Data preprocessing”), including a linear regression model to predict the gradient coefficients for the FVC curve based on the given data points and extracting additional features from the CT scans
2. Training of a base forecasting model based on the given feature data without the CT scans to establish a benchmark. For this, I used scikit-learn’s gradient boosting regressor
3. Merging the additional features from the lung CT scans, in my case lung volume and the percentage of fibrotic tissue in the patient’s lungs
4. Re-train the model from step 2 with the additional features and compare the outcome
5. Fine tune the model’s hyperparameters

3.2.2. Technologies & Libraries

The following technologies and libraries were used in my code

- Programming language: Python 3.8
- Pandas
- Numpy
- scikit-learn
- scikit-image
- pydicom library

3.3. Refinement

During the preprocessing of the CT images it became apparent that one code section is the most time consuming: Building the lung mask for every single CT scan, which I included below:

```
def make_lungmask(img):
    img = crop_borders(img)
    row_size= img.shape[0]
    col_size = img.shape[1]
    mean = np.mean(img)
    std = np.std(img)
    img = img-mean
    img = img/std

    # Find the average pixel value near the lungs to renormalize washed out images
    middle = img[int(col_size/5):int(col_size/5*4),int(row_size/5):int(row_size/5*4)]
    mean = np.mean(middle)
    max = np.max(img)
    min = np.min(img)

    # To improve threshold finding, I'm moving the underflow and overflow on the pixel spectrum
    img[img==max]=mean
    img[img==min]=mean
```

```
    # Using k-means to separate foreground (soft tissue / bone) and background (lung/air)
    kmeans = KMeans(n_clusters=2).fit(np.reshape(middle,[np.prod(middle.shape),1]))
    centers = sorted(kmeans.cluster_centers_.flatten())
    threshold = np.mean(centers)
    thresh_img = np.where(img<threshold,1.0,0.0)  # threshold the image

    # First erode away the finer elements, then dilate to include some of the pixels surrounding the
lung.
    # We don't want to accidentally clip the lung.

    eroded = morphology.erosion(thresh_img,np.ones([3,3]))
    dilation = morphology.dilation(eroded,np.ones([8,8]))

    labels = measure.label(dilation) # Different labels are displayed in different colors
    regions = measure.regionprops(labels)

    good_labels = []
    for prop in regions:
        B = prop.bbox
        height_check = row_size * 0.1 < B[2]-B[0] < row_size * 0.8
        width_check = col_size * 0.1 < B[3]-B[1] < col_size * 0.8
        position_row_check = B[0] > row_size * 0.2
        position_column_check = B[2] < col_size * 0.8
        if height_check and width_check and position_row_check and position_column_check:
            good_labels.append(prop.label)
        if len(good_labels) > 4:
            print('Lung mask contains more than 4 areas.')
            return False

    if len(good_labels) < 2:
        print('Lung mask contains less than 2 areas.')
        return False
    mask = np.ndarray([row_size,col_size],dtype=np.int8)
    mask[:] = 0

    #  After just the lungs are left, we do another large dilation
    #  in order to fill in and out the lung mask

    for N in good_labels:
        mask = mask + np.where(labels==N,1,0)
    mask = morphology.dilation(mask,np.ones([10,10])) # one last dilation
    if np.all(mask == 0):
        print('Black Image')
        return False
    applied_mask = mask*img
    tissue_only = np.where((applied_mask > 0) & (applied_mask < 0.9), 1, 0)

    return img, applied_mask, mask
```

The code could clearly be improved by

a) vectorizing all for loops and
b) converting the vectorized code into Pytorch tensors to process all of it on a GPU rather than on a CPU

Especially for the k-means algorithm, this would yield a much faster processing time that would help in a production environment.

Another refinement would be to integrate a k-fold cross-validation because the given sample size is rather small.

4. Results

In the last section I will summarize the result of my model and justify if the model is good enough to solve the problem.

4.1. Model Evaluation and Validation

As mentioned in section 2.4 the goal was to beat the “standard prediction” of mean (2690) and standard deviation (833) of the FVC of the training set. This yields a Laplace Log Likelihood Score (formula cf. 1.3 “Metrics”) of -8.023.

In the first iteration (only tabular data without CT image data), my model yields a score of -6.5799. With this model I predict a FVC for every week and the predicted standard deviation σ increases linearly the farther the prediction is away from the week in the given data point. For instance, if I have a given data point with a FVC of 3000 in week 12, then my prediction for week 12 is also 3000 with a confidence σ of 0, because it is completely certain that the value is 3000. For week 13, 14, 15, etc. the uncertainty - and therefore σ - increases week after week.

In the second iteration (including CT image data), my model yields a score of -6.5947 which is worse than the result of the first iteration. Practically, that would lead to the conclusion that a prediction based on the tabular data only leads to better results than if the CT image data is included.

Nevertheless, both iterations result in better predictions than just taking mean and standard deviation of the training set.

4.2. Justification

As described in the previous section my model gives better predictions to patients than just taking average values of all patients suffering from pulmonary fibrosis. However, there is a lot of room for improvement regarding analyzing and getting information from the CT images, because it is not plausible at all that more information (i.e., availability of a CT image) leads to worse results compared to a model with less information.
For the individual patient, this model provides a good approximation to her lung FVC decay, but at the end of the day there a many more factors of relevance than just sex, age and smoking status, for example overall fitness, other diseases, nutrition, activity, etc.