Prediction of iris type based on dimensions of their petals and sepals

Remigiusz Gębka
remigiusz.gebka@gmail.com

I. Introduction

In biology, especially in area of taxonomy, there are huge amount of problems with recognition of specific types, species. Many biologist are working on creating and recognizing features that are specific for this group of flowers or animals. These features, in science nomenclature, are called *keys*.

In this study, I've focused on dataset which gather information about irises - their petals and sepals. Experts gathered about 150 samples of irises - they have made some measurements and classified these samples to one of three species: Iris Setosa, Iris Versicolor and Iris Virginica. My task was to explore data and prepare few machine learning models to predict classification, based on petals and sepals of irises.

Details about the features extracted and the machine learning algorithms applied will be discussed in the following sections. Python and Jupyter Notebook was used to perform analyze.

II. Features and pre-processing

1. Data

Dataset contain 150 samples, for 50 samples per category – data is well balanced. Each row contains 4 measurements (petal width and length, sepal width and length) in separate columns. Last column is target feature – it's called Species. Whole files size is about 6KB.

2. Features

Dataset contain 5 columns (in raw data, space between words are replaced by dot):

- SepalLength– contains information about sepal length.
- SepalWidth – contains about sepal width
- PetalLength – contains about petal length
- PetalWidth – contains about petal width
- Species – contains category information (one of three values)

We should also know what exactly sepals and petals are. A ***sepal*** is a part of the flower of angiosperms (flowering plants). It's usually green, it's working mostly as a protection for flower in bud and as a support for petals. A ***petal*** is a modified leaves that surrounds the reproductive parts of flowers. They are often brightly colored or unusually shaped. Picture [Figure 1] below shows us build of flowers with petals and sepals signed – as example of Iris.

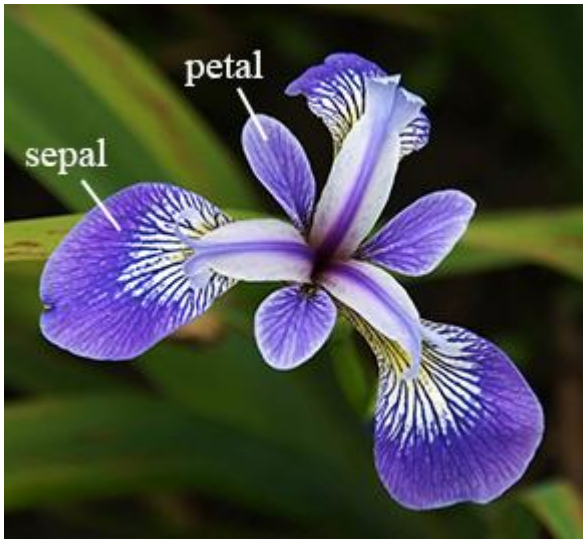

Figure 1
Sepal and petals signed on at Iris flower.
Source: https://www.math.umd.edu/~petersd/666/html/iris_with_labels.jpg

In Species column, we have 3 values: setosa, versicolor and virignica. These names correspond to names of species of Irises. Figure 2 shows us differences between these three species.

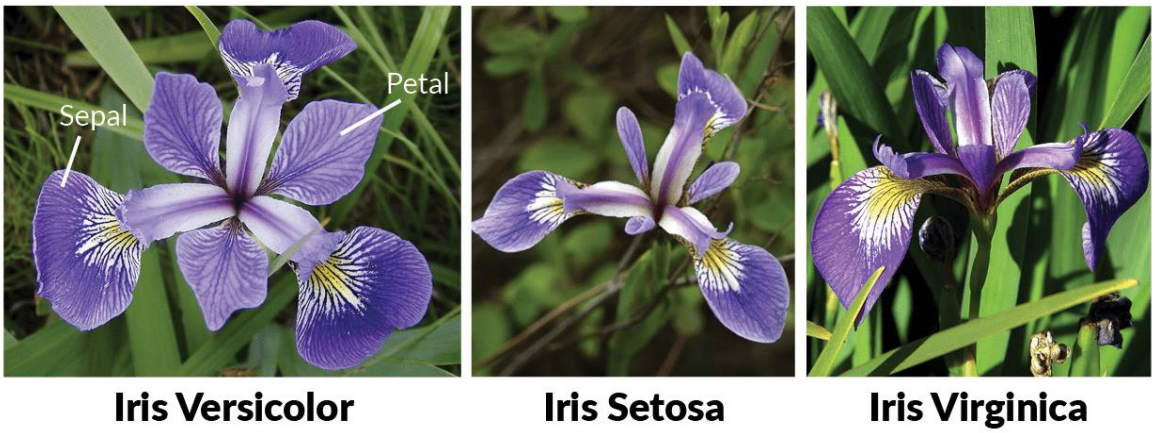

Figure 2
Three species of Irises.
Source: https://www.datacamp.com/community/tutorials/machine-learning-in-r

As we have some domain knowledge about, let's work on requirements of our data:

- All values in first 4 columns has to be float type, with value bigger than 0.
- Species column can contain only 3 unique values for model preparation
- Rows with missed values has to be dropped (then the data is not distorted) or have missing/incorrect values has to be filled with value that is consistent with the requirements (we are not loosing other, correct values – but filling these cells with e.g. mean values can cheat the results)

3. Cleaning

The first look into the data description shows some problems with it:

- One value, in SepalWidth column is missing
- One value, in SepalLength columns has negative value
- PetalWidth column is on object, not float type

Let's start with third problem with data – object type instead of float64 type. Looking into data gives information, that most of data is ready to be transformed into float type, but one of values are using comma instead of dot between two numbers. I've replaced that, and performed a transformation to float64 type by using included pandas methods.

To work with two next problems, we have to choose between dropping rows with missing values or using replacement for missing. For this study we will use both methods, so first data set (here and after called **Dropped Dataset**) will not contain rows with missed/incorrect information (so it will contain 148 samples, one from *setosa* category and one from *versicolor* category) and second one (here and after called **Average Dataset**) will be filled with average for all samples from column with missing data.

4. Analyze of correlation

We can now look into data. Let's start with visualization of our features using scatter plot for sepals parameters and petal parameters

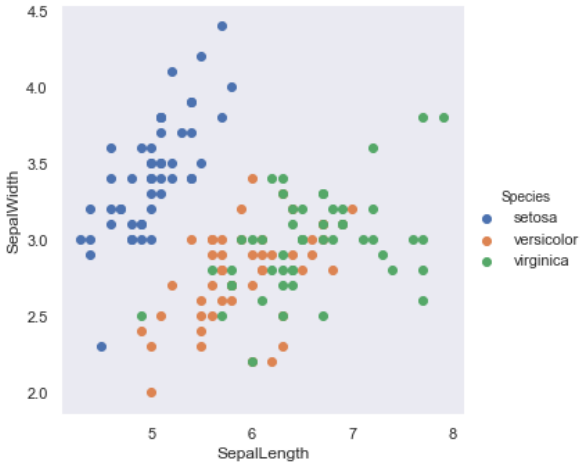

Figure 3
Scatter plot for Sepals using Dropped Dataset

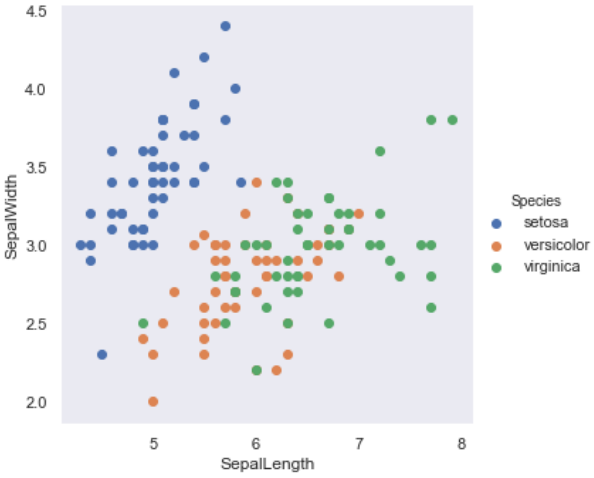

Figure 4
Scatter plot for Sepals using Average Dataset

As we can see, *setosa* samples have some separation from other two categories – with one outline, which is near *versicolor* samples. *Versicolor* and has very similar sepal length and width – and even if they take some part of area which belong mostly for them, they are mostly mixed. After looking into that, we can be almost sure that sepal width and length has correlation to category, but there will be problem with classifying only these two features.

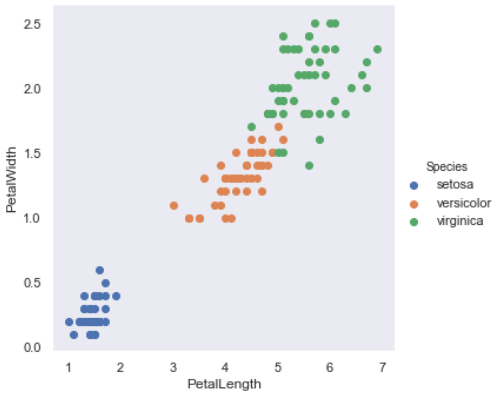

Figure 5
Scatter plot for Petals using Dropped Dataset

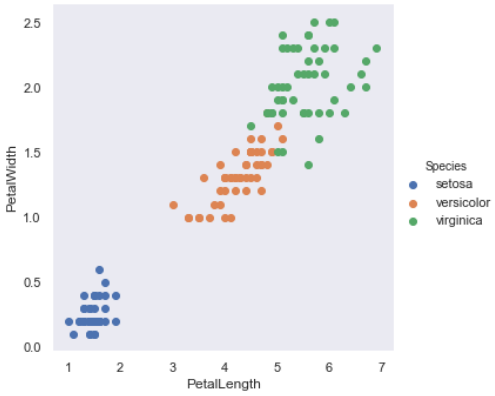

Figure 6
Scatter plot for Petals using Average Dataset

At Figure 5 and Figure 6 - we see Petals features samples visualized on scatter plots – and we see, that here we have good better correlation to category. Setosa has again separated area – and other two has now better samples separation, and even if they have some mixed samples, amount of them is really low and values are not very outlined from other values. We will have probably better predictions using these two features.

Same conclusions to data can give us look into density plots – Figure 7 and Figure 8 shows visualization of density using violin plots.

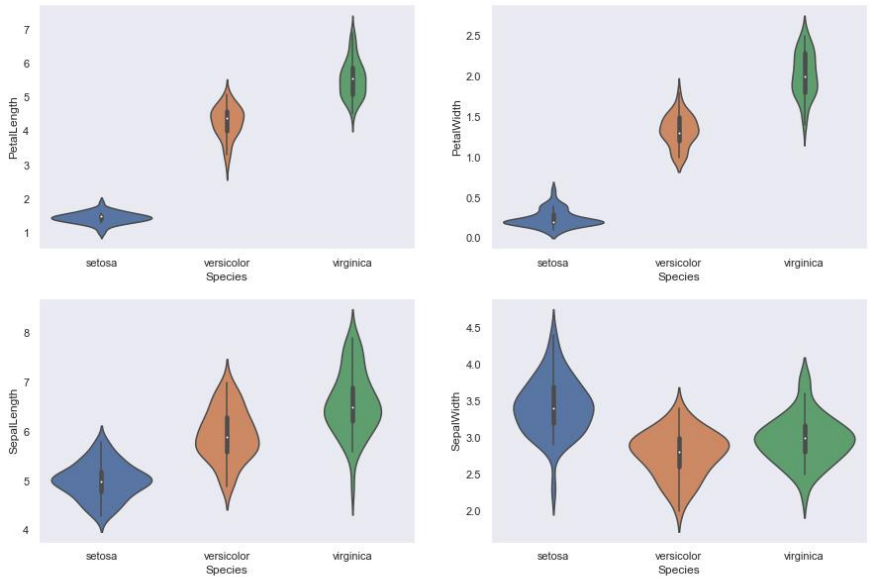

Figure 7
Violin plot for Dropped Dataset for all features

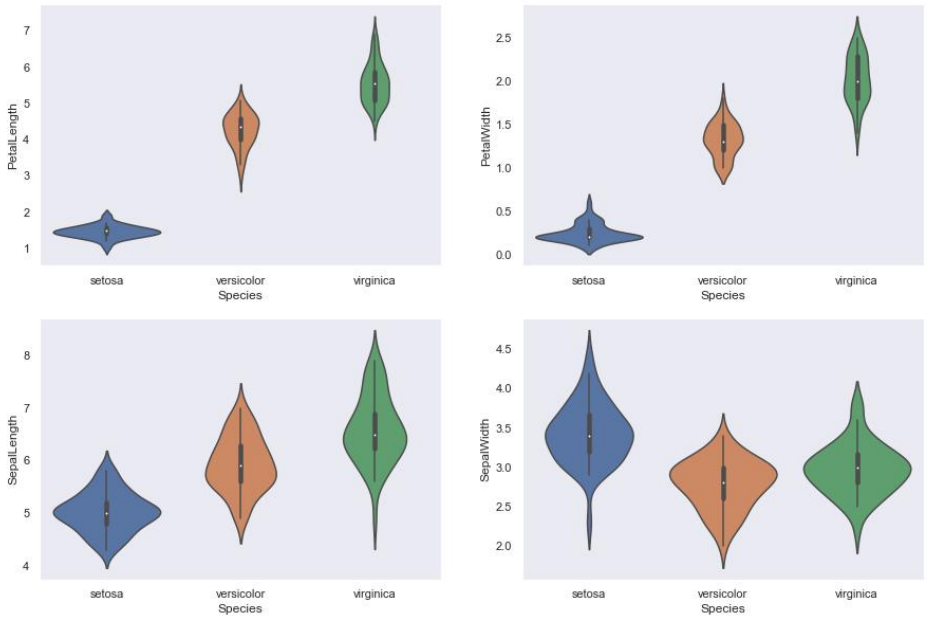

Figure 8
Violin plot for Average Dataset for all features

Let's take a wider look into data – we have 4 dimensions here, and scatterplot can show us only two of them at one time – so we need something better to visualize data. There are some methods to do that, but we need only two of them at this moment: Parallel Coordinates and Radviz. First of them show all features in one plot, which use one column for each feature. The second one plots a N-dimensional data set into a simple 2D space where the influence of each dimension can be interpreted as a balance between the influence of all dimensions.

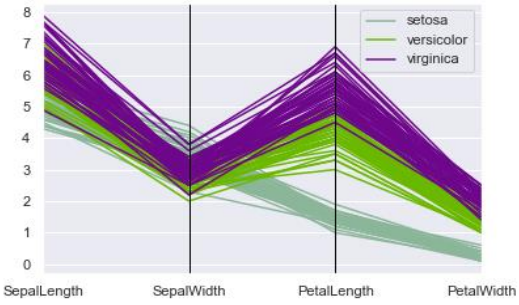

Figure 9
Parallel coordinates plot using Dropped Dataset

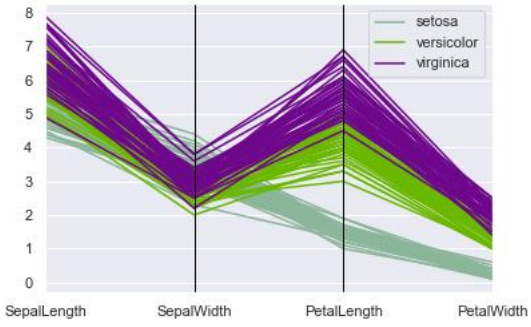

Figure 10
Parallel coordinates plot using Average Dataset

Figure 11
Radviz using Average Dataset

Figure 12
Radviz using Dropped Dataset

As we can see, *setosa* samples will not have bigger problems with classification of them – bigger problem can be with two remaining categories - they've mixed area and precision in their case has to be slightly lower.

Take a look into correlation between features and correlation between features and target feature.

	SepalLength	**SepalWidth**	**PetalLength**	**PetalWidth**
SepalLength	1.000000	-0.117009	0.871145	0.815815
SepalWidth	-0.117009	1.000000	-0.429040	-0.367709
PetalLength	0.871145	-0.429040	1.000000	0.962910
PetalWidth	0.815815	-0.367709	0.962910	1.000000

Table 1
Correlation between features in Dropped Dataset

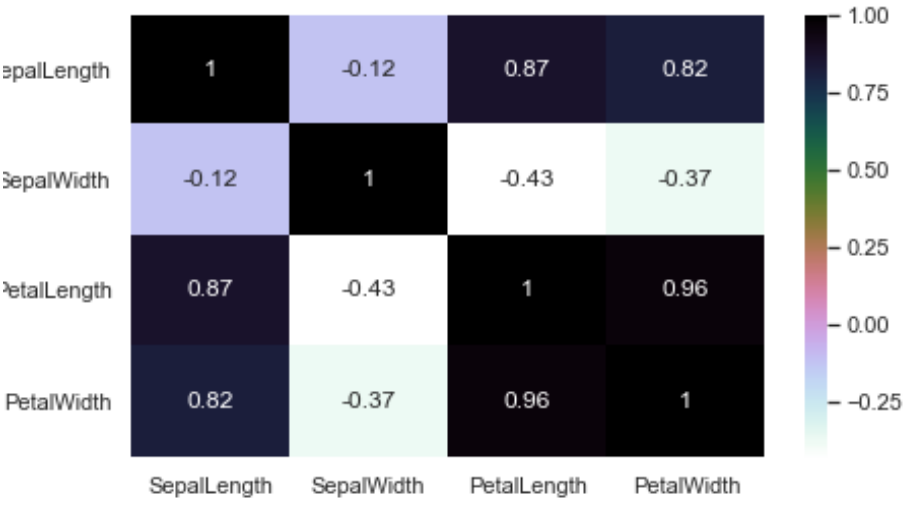

Figure 13
Heatmap for correlation in Dropped Dataset

	SepalLength	SepalWidth	PetalLength	PetalWidth
SepalLength	1.000000	-0.116711	0.867459	0.811142
SepalWidth	-0.116711	1.000000	-0.432108	-0.371657
PetalLength	0.867459	-0.432108	1.000000	0.962865
PetalWidth	0.811142	-0.371657	0.962865	1.000000

Table 2
Correlation between features in Average Dataset

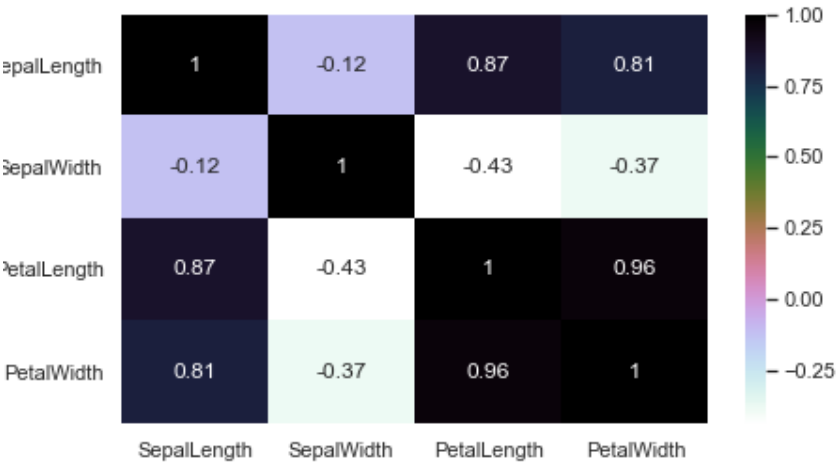

Figure 14
Heatmap for correlation in Average Dataset

Correlation between datasets have only few small differences – and they are below 0.05. Interesting thing, that we can see there, is the fact that Sepal features have really low correlation between them, in opposite to Petal features – we can see there strong correlation. That says us a lot about usability of these features in model preparation – using only Petal features should give us better scores than using Sepal features. Now we should check correlation between features and target feature – to do that, we can do two things: we can use LabelEncoding to encode categorical variable to number or use One-Way Anova test and check F-statistic and p-value for each feature.

	Encoded category correlation in Average Dataset	Encoded category correlation in Dropped Dataset	F-statistic	P-value
SepalLength	0.77	0.78	114.07	1.24e-30
SepalWidth	-0.42	-0.42	47.7	1.07e-16
PetalLength	0.94	0.94	1180	2.85e-91
PetalWidth	0.95	0.95	960	4.17e-85

Table 3
Correlation to categories using both methods

Correlation in that case is just confirmation of conclusion made based on all visualizations – all features have impact on categories (p-value confirms correlation with F-statistic).

III. Models

To work with models, we have to prepare training and test set, to train and validate model. I've used sklearn library to do that. Our data had ratio: 70% as a training set, 30% as a test set. I've also used a random_state parameter – to have possibility to reproduce tests.
As models I've used k-Nearest Neighbors (k-NN) algorithm, Support Vector Machine (SVM) and Decision Trees. To validate model, I've used few metrics:

- Accuracy - Accuracy is the most intuitive performance measure and it is simply a ratio of correctly predicted observation to the total observations. (TP+TN/TP+FP+FN+TN)
- Precision - Precision is the ratio of correctly predicted positive observations to the total predicted positive observations. (TP/TP+FP)
- Recall - Recall is the ratio of correctly predicted positive observations to the all observations in actual class - yes. (TP/TP+FN)

Every model after preparation & validation was saved to disc – to future use.

Due to fact that we have two datasets to test – all models were created two times, one with first data set and next with second one, to compare their quality.

1. *k*-NN

k-NN works very good for this task – with one of datasets and parameters, I've acquired theoretical 0.97 accuracy – with this specific test set. As a parameter, that I should test, I've taken a number of neighbors.

Table with Dropped dataset results:

Parameter value	Accuracy	Precision	Recall
3	0.95	0.95	0.95
5	0.95	0.95	0.95
7	0.93	0.93	0.93
11	0.97	0.97	0.97

Table with Average dataset results:

Parameter value	Accuracy	Precision	Recall
3	0.93	0.93	0.93
5	0.91	0.91	0.91
7	0.93.	0.93	0.93
11	0.91	0.91	0.93

2. SVM

SVM works impressive with linear parameter – we acquire 1.0 theorical accuracy with F1 score (as a combination of recall and precision). Parameter that I've changed there was kernel type. For average dataset – linear works a little worse, but rbf works better in that set.

Table with Dropped dataset results:

Parameter	Accuracy	Precision	Recall
Rbf	0.97	0.97	0.97
Linear	1.0	1.0	1.0
Poly	0.97	0.97	0.97
Sigmoid	0.26	0.26	0.26

Table with Average dataset results:

Parameter	Accuracy	Precision	Recall
Rbf	0.95	0.95	0.95
Linear	0.93	0.93	0.93
Poly	0.88	0.88	0.88
Sigmoid	0.26	0.26	0.26

3. Decision Trees

Decision Trees has good scores – and very stable in most cases. I've worked there with two parameters changes max_depth (here and after as **mD**) and with min_sample_leaf (here and after as **mSL**).

Table with Dropped dataset results:

Parameter	Accuracy	Precision	Recall
mD: default, mSL: 1	0.97	0.97	0.97
mD: 2, mSL: 1	0.95	0.95	0.95
mD: 3, mSL: 1	0.97	0.97	0.97
mD: 4, mSL: 1	0.97	0.97	0.97
mD: 3, mSL: 2	0.97	0.97	0.97
mD: 3, mSL: 3	0.97	0.97	0.97
mD: 3, mSL: 4	0.97	0.97	0.97

Table with Average dataset results:

Parameter	Accuracy	Precision	Recall
mD: default, mSL: 1	0.91	0.91	0.91
mD: 2, mSL: 1	0.88	0.88	0.88
mD: 3, mSL: 1	0.86	0.88	0.88
mD: 4, mSL: 1	0.86	0.86	0.86
mD: 3, mSL: 2	0.88	0.88	0.88
mD: 3, mSL: 3	0.88	0.88	0.88
mD: 3, mSL: 4	0.88	0.88	0.88

4. Tests with only Petal and with Sepal features

Just to test and confirm visualization and correlation between variables – I want to show a table with scores acquired by using only Sepal features and Petal features.

Algorithm	Dropped data, sepals	Average data, sepals	Dropped data, petals	Average data, petals
kNN	0.75	0.77	0.97	0.91
SVM	0.8	0.71	0.97	0.93
Decision Tree	0.64	0.62	0.97	0.88

That results confirm my previous thoughts about correlation.

We can also see, in that case – using drop instead of filling values give us better results.

IV. Analyze of missing data

As you can read in previous section, using global mean works, in that case, worse than dropping missing/incorrect values. I've check this – below you can find visualization of this samples as a separated categories with missing_ prefix.

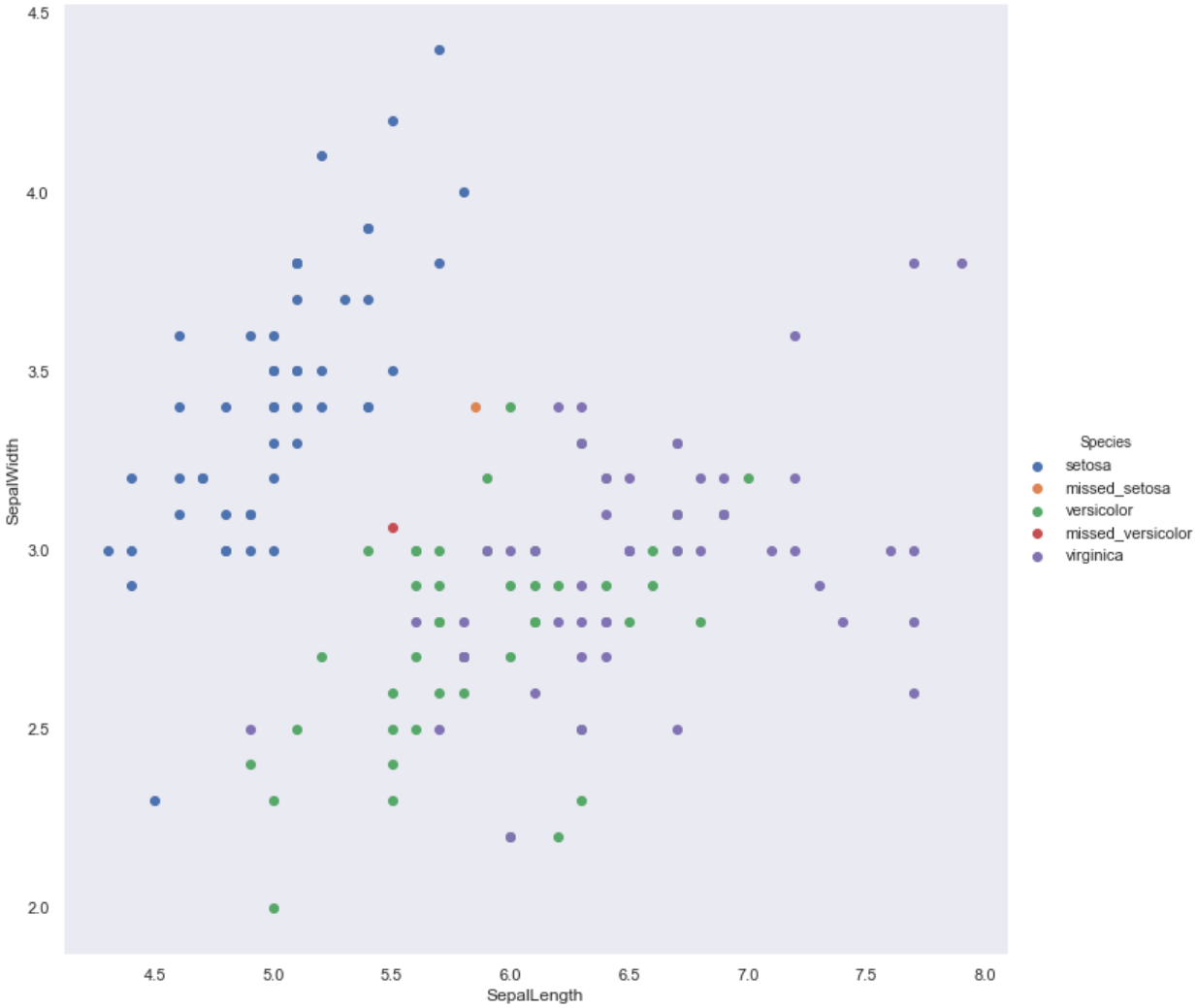

Figure 15
Scatter plot with marked missing samples

As you can see at Figure 15, using global mean makes sample from missed_versicolor – red one - (versicolor originally) a border sample – not an outline, but still it's on a border of area taken by versicolor. Second one – orange – originally setosa, makes this sample an outline from setosa. We know now why using the global mean is bad idea. What about local (category) mean using? Take a look into Figure 16.

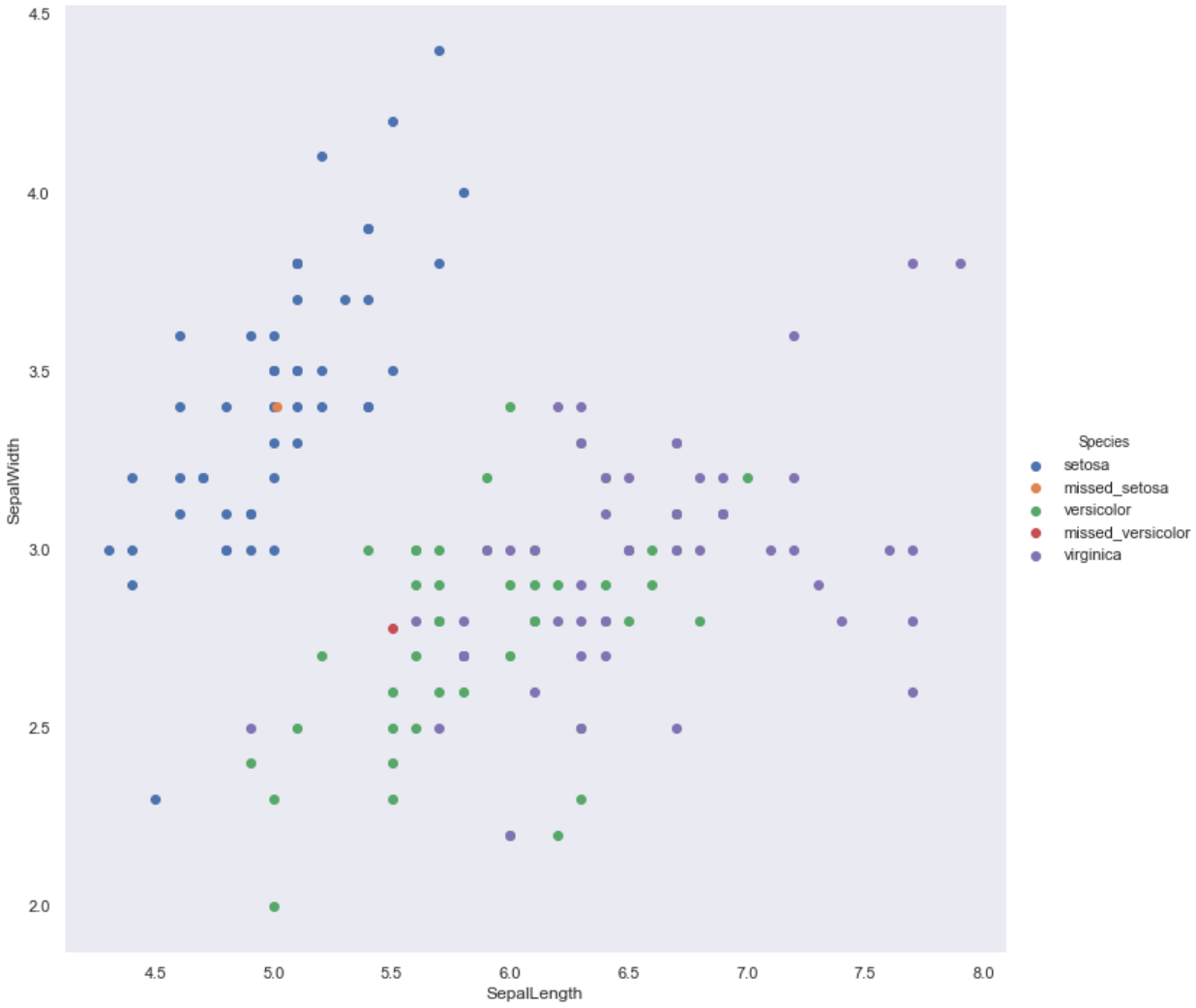

Figure 16
Scatter plot with marked missed samples – now filled with category mean

We can see, missed samples are now in core part of rest samples – that can improve quality of models or just prevent noise in the data.

V. Creating additional features

For this project, like for others, feature selection and engineering is a key to success. In data we have 4 columns with data – but maybe using only part of them, or using just ratios of dimensions ratios of selected flower elements will generate better score.

For this task, let's define two new columns:

- Ratio_sepal, which is calculated by dividing SepalLength by SepalWidth
- Ratio_petal, which is calculated by dividing PetalLength by PetalWidth

Take a look into corraltion matrix (using label encoder):

```
SepalLength        0.781994
```

```
SepalWidth       -0.429357
PetalLength       0.949035
PetalWidth        0.956547
ratio_sepal       0.779237
ratio_petal      -0.679061
```

New features have correlation to target variable, even higher than one of basic features – but still lower than rest features. As we can see on Figure 17, these two features have same problem as basic features – versicolor and virignica species are mixed.

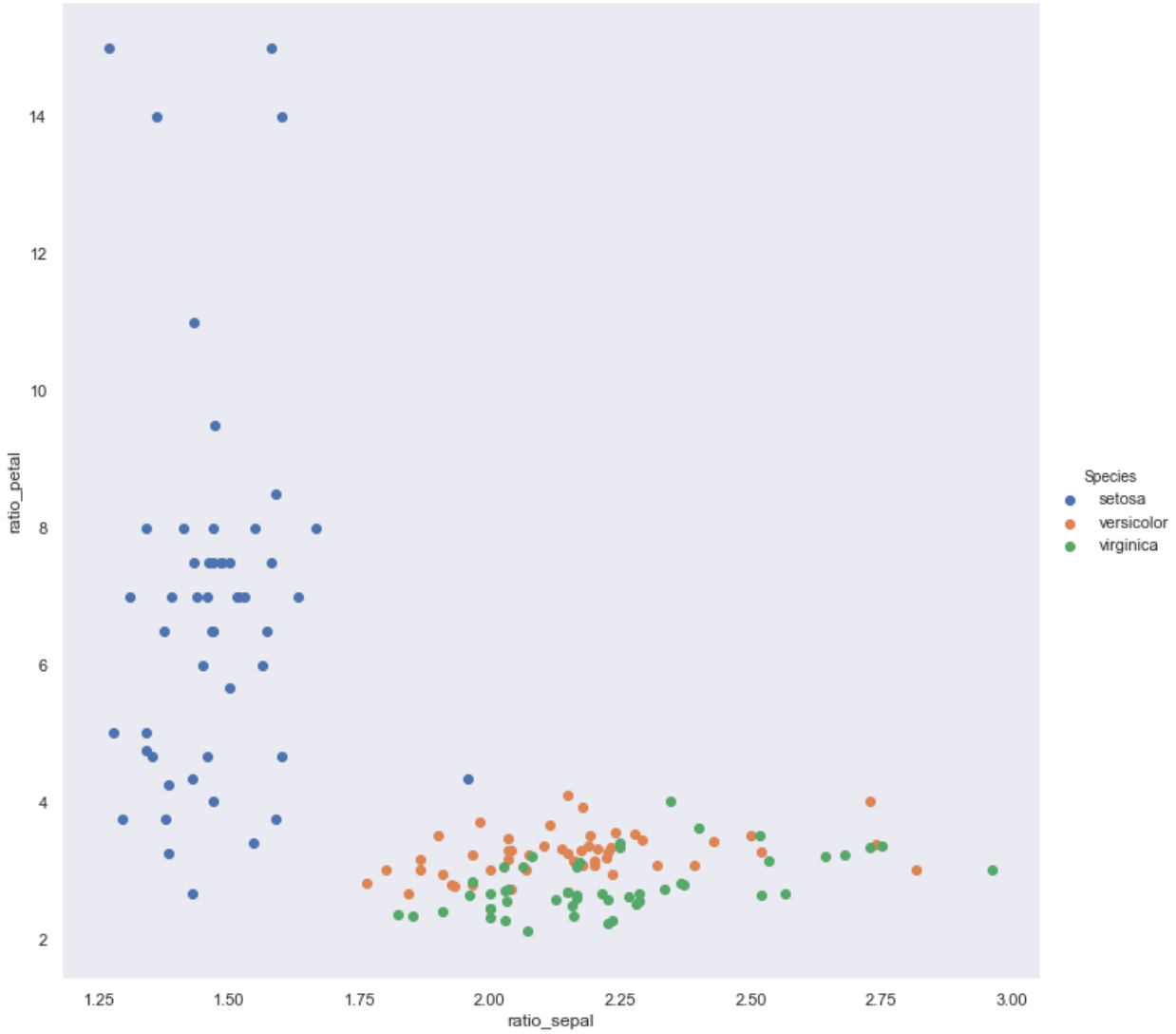

Figure 17
New feature visualization using scatter plot

Now we should take a look into direct scores of models, which are represented on Table 4.

Model	*k*-NN accuracy
Reference model	0.93
Basic + new features	0.93
Basic + sepal ratio	0.91
Basic + petal ratio	0.93
New features only	0.84

As we can see, new features presented above are not improving model accuracy.

VI. Conclusions

As shown in this report there are enough information present in dataset to identify a type of iris based on petals and sepals parameters. SVM model looks encouraging enough to show that we can use it in production environment to help client in iris classification.

In the future, more features combination can be tested, to reduce complexity of data, to reduce dimensions, like also there are still many algorithms which are not tested now – e. g. random forest, logistic regression. The problem also lies in decision what we should do with missing or incorrect data – in that dataset dropping them was a good idea, but with another data maybe another type of action can be better.