Project Report

GitHub URL

https://github.com/riju-s/UCDPA_Riju-Sathyan

Abstract

The report seeks to demonstrate the application of skills gained within Python to successfully import, clean, manipulate and visualise results – thereby enabling insights to be highlighted through the journey. The first part of the project uses English Premier League (EPL) table datasets sourced from Kaggle and Wikipedia. The latter dataset enabling us to validate data in the former dataset, prior to any analysis or visualisation. The second part of the project uses Diabetes data, sourced from Kaggle, to perform classification type supervised machine learning to see if the eight feature variables can accurately predict the target variable.

Introduction

The EPL datasets were chosen because to avoid the need for an SMEs to assist in your work or extensive research. Given that I am an avid follower of the EPL since its inception in the early 1990s, it seemed like a logical dataset to demonstrate learnings with a domain I was familiar with. However, this dataset has drawbacks for machine learning, given the level of interdependencies between records. i.e., a win for one team is a loss for another, a goal conceded for one team is goal scored for another, etc.

Therefore, I selected a more machine learning friendly dataset that did not have the same level of complex interdependency issues. Given diabetes is more likely in South-East Asian communities, my heritage, I thought it would be interesting to test the thesis – using machine learning, could a set of eight feature variables (noted below) be used to accurately predict whether an individual was a diabetic.

Dataset

Description	Source/File Name	Source Link
(a) Champion League History - UK Teams from 1955 to 2022	Wiki page table	https://en.wikipedia.org/wiki/English_football_clubs_in_international_competitions
(b) English Premier League Final Tables for 21 Seasons	EPL_standings_2000-2021.csv	https://www.kaggle.com/datasets/quadeer15sh/premier-league-standings-11-seasons-20102021
(c) Diabetes data on female patients (min 21 years old) of Pima Indian heritage	diabetes.csv	https://www.kaggle.com/code/mathchi/diagnostic-a-patient-has-diabetes/data

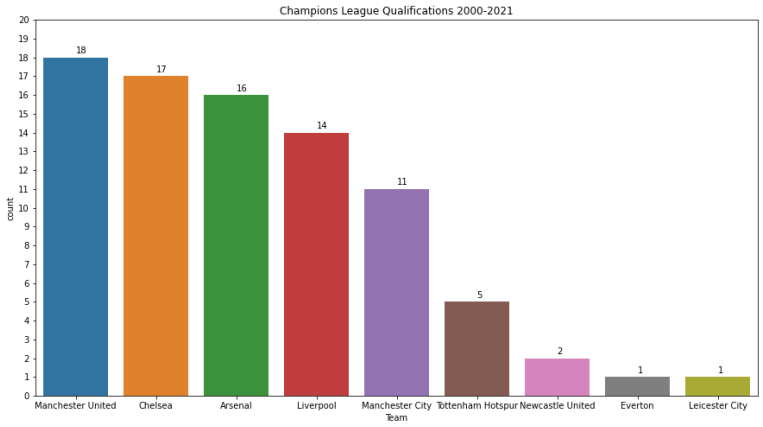
Implementation Process

EPL Datasets

- Imported the necessary modules for this exercise.
- EPL Dataset (a): Webscraped table from a Wikipedia web page using pandas **read_html** function and stored the results within a **pandas** dataframe named *df_wiki*.
- Located the desired dataset as index=3 within the df_wiki, and visually confirmed results.
- Reviewed the integrity of the data by using **info** and **describe** functions
- Replaced the character '—" with '-" on the *Season* column, with the **str.replace** function to align format with the EPL dataset (b).
- **Regex** in conjunction with **str.contains** used to check referential integrity of *Season* column, with results stored as Boolean values in a new column *DQ_Season*. Exceptions where reviewed and updated.
- A new dataframe, *df_seasons,* was created with sorted unique Champion Leagues (CL) seasons. The function **shift** was used to create a new column lagged by one row, to reflect the Premier League qualifying season, which the prior season.
- A **dictionary**, sm_d, was created and then populated by iterating through *df_seasons*, to create a key-item mapping between Champion League season and qualifying Premier League season.
- The **zip** function was used to create a new dataframe, *df_English_teams_qualified*, from two columns from different dataframes.
- A new column *Season_PLq* was created and populated through **iterating** through all records within the dataframe. Item value where stored from the key (based CL Season) within the *sm_d* **dictionary**.

- EPL Dataset (b): Imported csv file as a **pandas** data frame, *df_EPLs.* A total of 440 records.
- 20 records were dropped where the columns *Season*='2021-22' using **~**. This is needed because dataset (A) does not contain this period.
- Reviewed the integrity of the data by using info and describe functions
- **Regex** in conjunction with **str.contains** used to find champion league qualification from text within the *Qualification or relegation* column, with results stored as Boolean values in a new column *CLQ_chk*. Exceptions where reviewed and updated.
- Two data frames created that containing unique team names from both datasets (A) and (B).
- The two datasets were **merged** using a **right join** and a **Boolean** check made on whether the team names were the same across both team name columns. This exercise was conducted to ensure consistency in Team names.
- Exception were reviewed, but no updates were necessary
- Created the data frame, *df_final_table*, by merging df_EPLs and *df_English_teams_qualified* vaia left join on both *Season* and *Team* columns
- Revalidated that no Teams were unmatched as a result of this merge.
- This means that the teams and seasons on dataset (b) have been cross-checked and validated with an alternative data source, i.e. dataset (b)
- Removed unwanted columns using the **drop** function.
- Used the final dataset to start analysis.

- Created a **countplot** using **Seaborn** and **matplotlib** to show which teams has qualified for Champion League the most time (**sorted descending**) over the 22 year period.



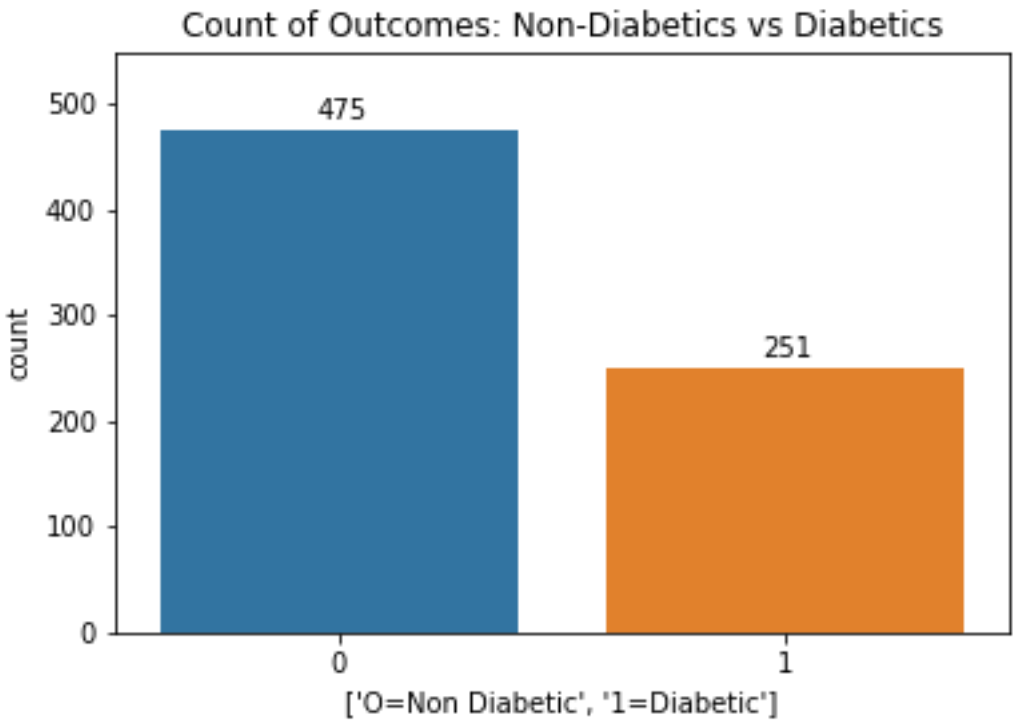
- Two **user-defined functions** were created – *get_teams()* and *get_history().* The full description of the functions can be found using **__doc__**.
- The functions were used to returns results and a basic iteration was used to demonstrate the returned values of a selected range of indices.

Diabetes dataset

- Imported the additional modules required for this exercise.
- Imported csv dataset into pandas dataframe using **read_csv**.
- Here is a summary of each of the nine variables/columns:

Variable	Description
Pregnancies	Number of times pregnant
Glucose	Plasma glucose concentration a 2 hours in an oral glucose tolerance test
BloodPressure	Diastolic blood pressure (mm Hg)
SkinThickness	Triceps skin fold thickness (mm)
Insulin	2-Hour serum insulin (mu U/ml)
BMI	Body mass index (weight in kg/(height in m)^2)
DiabetesPedigreeFunction	Diabetes pedigree function
Age	Age (years)
Outcome	Class variable (0 or 1)
*SOURCE: Mehmet Akturk – see references	

- Reviewed the integrity of the data by using **info** and **describe**. Alongside this, the data was visually reviewed using **seaborn boxplots**, for each variable.
- This review highlighted the following:
    a) No null values



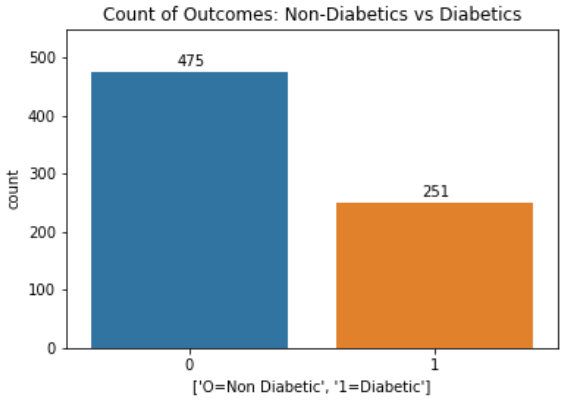
    b) All columns contained 768 records
    c) Minimum age was confirmed as 21
    d) *Glucose*, *BloodPressure*, *SkinThickness*, *Insulin* and *BMI* all contained zero values, which clearly is erroneous and thereby skewing the statistical metrics.
    e) Highly likely erroneous outlier values in *Pregnancies*, *Glucose*, *BloodPressure*, *SkinThickness*, *Insulin*, *BMI* and the *DiabetesPedigreeFunction*.
- Point [d] was addressed by converting all zero values in the impacted columns to numpy NaN, recalculating the column medians, and then replacing the zero values with the respective updated median. This approach avoided the need to drop a fair number of records.

- Point [e] was addressed by removing outlier records. The definition of an outlier used was 0.5 percentile (upper and lower). The respective records were removed from the dataset leaving 726 records, ie 42 records dropped.
- The final data set was reviewed again using **info** and **describe**. Alongside this, the data was visually reviewed using **seaborn boxplots**, for each variable.
- A countplot was used to show split between non-diabetics and diabetics, where *Outcome* is zero and one respectively. In addition, the **describe** function was also used to observes the differences between the two Outcome types.
- A correlation matrix heatmap was used to highlight the relationships between the variables
- For the top three correlation values, further analysis was conducted.
- The relationships between Age and Pregnancies, and BMI and Skin Thickness was reviewed using seaborn **regplots** for each Outcome time. This showed the linear regression model fits between non-diabetic and diabetic groups. Additionally, a seaborn scatterplot was reviewed across the whole outcome population with hue set to differentiate outcome groups. All three plots were created as subplots with a common x and y axis to enable easier side-by-side comparison.
- Given the third highest correlation was between Outcome (target variable) and Glucose, different analysis was undertaken. A seaborn **displot** was used to plot the distribution of non-diabetics and diabetics onto a single plot, with a kernel density estimates (KDEs) added to see the shape of the distribution. The respective means for each sub-group was also added to the plot.
- Visually the profile of glucose levels in diabetics looked materially lower when compared to the glucose levels in diabetics.
- To test statistically, a t-test was conducted on the two sub-datasets (glucose levels in non-diabetics vs glucose levels diabetics). The null hypothesis of both sub-datasets having the same mean was tested with a confidence level of 99%. The null hypothesis was rejected, thereby suggesting that the means are statistically significantly different. This was all achieved using the **scipy.stats** module and the **ttest_ind** function.
- Two dataframes were created with the feature variables and target variable. Each dataset was then split into a training and test groups, using a 40% split ratio and allocated a random state (for consistent repeatability)
- A range of matching learning classification models were run on the datasets to see if the feature variables could accuracy predict the target variables. The four classification algorithms used were Random Forests (RDF), Decision Tree (DTC), Support Vector Machines (SVM) and K-Nearest (KNN).
- A loop was created to fit all the model (with default values) and then to predict the test target variable. The model accuracy was then stored in the dictionary variable *model_results*.
- For the two models (RDF & DTC), feature importance was also calculated.
- A loop was created to fit a range of parameters to each respective model via GridSearchCV to undertake hyperparameter tuning and then to predict the test target variable. The model accuracy was then also stored in the dictionary variable *model_results*.
- A key-item was added to the dictionary to store the difference between model accuracy for each model, before and after hyperparameter tuning.
- The dictionary of dictionaries was then converted into a pandas dataframe and transposed. This produced the final results table.

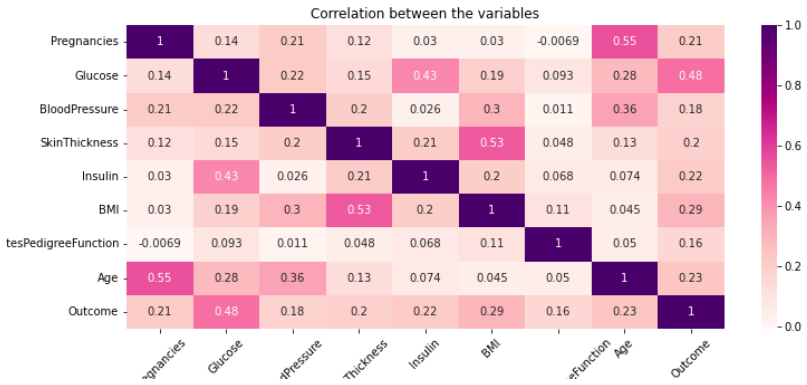
Results

Zero values contained in diabetes.csv		
Column	Zero Values Count	% of Dataset
Pregnancies	0	0.0
Glucose	5	0.7
BloodPressure	35	4.6
SkinThickness	227	29.6
Insulin	374	48.7
BMI	11	1.4
DiabetesPedigreeFunction	0	0.0
Age	0	0.0

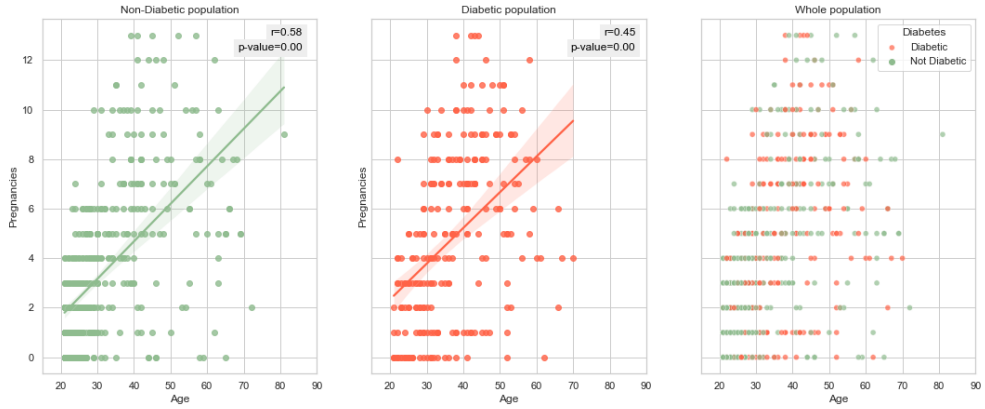
The table shows the number of erroneous zero values contained in the diabetes dataset. 48.7% and 29.6% of Insulin and SkinThickness data collected, respectively, was impacted by this issue.



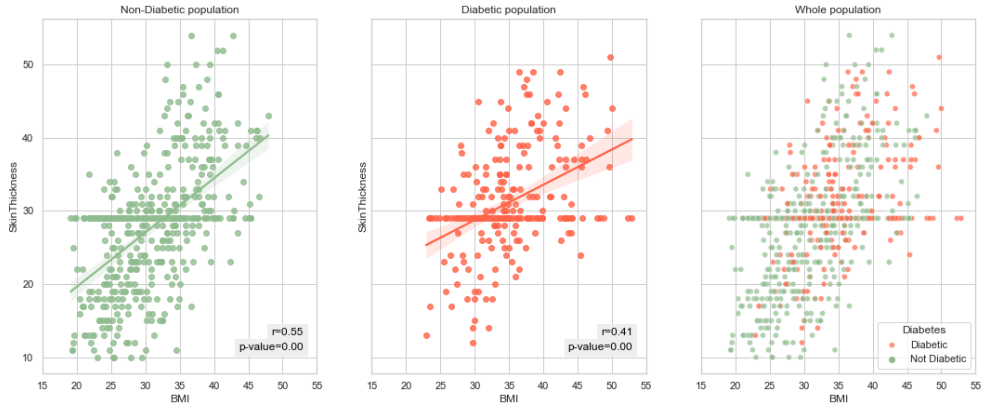
The plot shows the count of 475 non-diabetics (65.4%) and 251 diabetics (34.6%) within the final dataset.



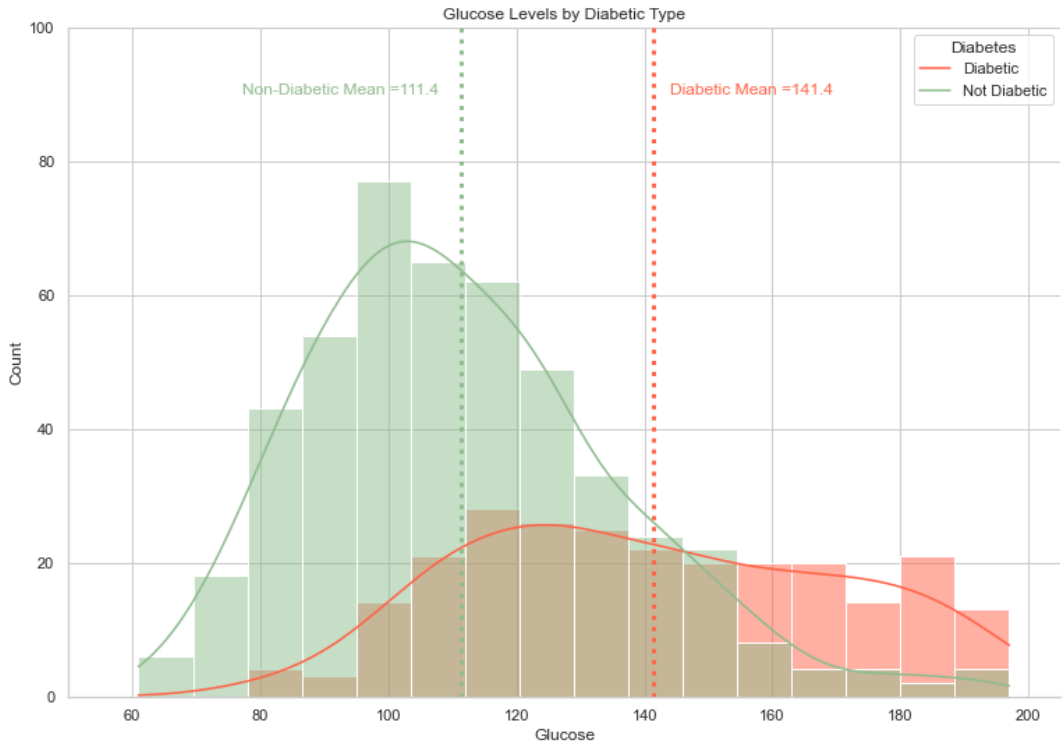
The correlation matrix heatmap highlights the correlation between all variables in the dataset. The strongest correlation exists between Age & Pregnancies, BMI & Skin Thickness and Glucose & Outcome. All three correlations were positive.



The first two plots show the different correlations for both non-diabetics and diabetics between Age and Pregnancies variables. Both plots highlight a positive correlation, with the stronger correlation found in the non-diabetic population. The p-values suggest that the correlations are valid. The final scatter plot highlights the observations on a single plot.



The first two plots show the different correlations for both non-diabetics and diabetics between BMI and Skinthickness variables. Both plots highlight a positive correlation, with the stronger correlation found in the non-diabetic population. The p-values suggest that the correlations are valid. The final scatter plot highlights the observations on a single plot.



The displot show the distribution of glucose levels in the non-diabetic and diabetic populations. The kernel density estimates highlight the shape of each distribution and the respective means highlighted. The mean glucose levels in diabetics is higher (141.4 units) than in non-diabetics (111.4 units). The student t-test confirmed that the differences in mean were significant to a 99% confidence level.

	pre hypertuning/default accuracy score	post hypertuning accuracy score	improvement in accuracy score
RDF	0.728522	0.735395	0.006873
DTC	0.690722	0.652921	-0.037801
SVM	0.725086	0.725086	0.000000
KNN	0.694158	0.714777	0.020619

The table highlights the model accuracy scores for each classification algorithm. When run with default parameters, accuracy ranged between 69.1% and 72.9%. Following hyperparameter tuning, two algorithms, K-Nearest (KNN) and Random Forest (RDF) showed an improvement in model accuracy by +2.1% and +0.1% respectively. The algorithm Support Vector Machines (SVM) remain unchanged post hyperparameter tuning. Finally, the Decision Tree (DTC) algorithm model accuracy worsened by -3.8% after hyperparameter tuning.

Overall, the highest model accuracy score was achieved by RDF with 73.5%. The parameters combination used to achieve this result was:

```
RDF Best Hyper Parameters:
 {'criterion': 'gini', 'min_samples_leaf': 2, 'min_samples_split': 3, 'n_estimators': 90, 'n_jobs': -1, 'random_state': 999}
RDF Accuracy: 0.7353951890034365
```

Feature Importance: RDF	
Glucose	0.303646
BMI	0.138453
Age	0.124576
DiabetesPedigreeFunction	0.110892
BloodPressure	0.090488
Insulin	0.084993
Pregnancies	0.081874
SkinThickness	0.065078

Feature Importance: DTC	
Glucose	0.371929
BMI	0.130469
Age	0.105302
Pregnancies	0.100608
BloodPressure	0.089122
Insulin	0.074707
SkinThickness	0.066267
DiabetesPedigreeFunction	0.061594

The feature importance tables highlighted, for RDF and DTC algorithms, glucose levels is the single most important variable in predicting the outcome variable.

Insights

- Improve data collection processes on Insulin and SkinThickness data.

- There exists a notable positive correlation between Age and Pregnancies. However, there is a stronger positive correlation between Age and Pregnancies in non-diabetics when compared to diabetics.

- There exists a notable positive correlation between BMI and SkinThickness. However, there is a stronger positive correlation between Age and Pregnancies in non-diabetics when compared to diabetics.

- There is a statistically significance difference between in average glucose levels in diabetics compared to non-diabetics.

- Of the eight feature variables Glucose levels is the most important feature in predicting diabetes in this population.

- Random Forest (RDF) post-hyperparameter tuning yielded the best model accuracy score of 73.5%.

References

1 Mehmet Akturk: Diagnostic a patient has DIABETES
https://www.kaggle.com/code/mathchi/diagnostic-a-patient-has-diabetes

2 Kumagawa An - Decision Tree: Hyperparameter & Visualizing
https://www.kaggle.com/code/ankumagawa/decision-tree-hyperparameter-visualizing