Statistical Learning for Big Data

BEBB5001

Semester Project

Abdur Rehman Anwar Qureshi

198005101
MS CS

Department of Computer Science,
Institute of Informatics,
GAZI UNIVERSITY, ANKARA

Data Understanding

The problem is related to diabetes prediction by using machine learning techniques. Diabetes is a disease in which patients' blood glucose, or blood sugar, levels are too high. The dataset is downloaded by **Kaggle**[1] and **Github**[2] as references are mentioned in the reference section which is given at the bottom. This dataset is also available on the **UCI Machine Learning Repository**[3]. The objective of this dataset is to diagnostically predict whether or not a patient has diabetes, based on certain diagnostic measurements included in the dataset. All patients are female and their age is at least 21 in this dataset. The dataset consists of 9 medical predictable variables i.e. pregnancy, glucose, blood pressure, skin thickness, insulin, BMI, diabetes per degree function, and age. And one target value i.e outcome. The outcome has been shown as 0 or 1.

Data Preparation

This dataset consists of 768 tuples of different patients. Each tuple has 8 different features and one outcome. This is the target class which is 0 or 1. All features values are numeric form. I did normalize feature values with the help of the **z-score data standardization** technique as features variables transforming and then used for diabetic prediction. Before moving ahead, I would like to explain the z-score data standardization technique. It is also known as z-score normalization. The formal definition of z-score normalization is described below.

Firstly, we need to compute the mean of particular feature variable by using a formula as follows:

$$\mu_f = \frac{1}{n}\sum_{i=1}^{n} x_{if} \quad . (1)$$

After that, the standard deviation is calculated by using the mean value of the particular feature and all values of a similar feature as follows:

$$\sigma_f = \sqrt{\frac{\sum_{i=1}^{n}(x_{if} - \mu_f)^2}{n-1}} \quad . (2)$$

---

[1] Kaggle is online community of data science and machine learning practitioners. https://www.kaggle.com/ (accessed on 20 March 2020)

[2] Github is US-based software company which provides hosting software development version control by using Git. https://github.com/ (accessed on 21 March 2020)

[3] UCI Machine Learning Repository is a collection of dataset which are used by machine learning community. https://archive.ics.uci.edu/ml/index.php (accessed on 21 March 2020)

Finally, the z-score normalization or z-score data standardization can be computed by using equation 1 and equation 2 as follows:

$$z(x_{if}) = \frac{x_{if} - u_f}{\sigma_f}. \quad (3)$$

The z-score formula is equation 3 has been applied to each feature value of the diabatic dataset to performed data transformation for prediction models.

Data Visualization

I have designed and developed this assignment in Jupyter Notebook (IDE)[4]. As a consequence, visualized the dataset before and after data standardization in Fig 1 and Fig 2 respectively.

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age
0	6	148	72	35	0	33.6	0.627	50
1	1	85	66	29	0	26.6	0.351	31
2	8	183	64	0	0	23.3	0.672	32
3	1	89	66	23	94	28.1	0.167	21
4	0	137	40	35	168	43.1	2.288	33
...	...	...	...	...	...	...	...	...
763	10	101	76	48	180	32.9	0.171	63
764	2	122	70	27	0	36.8	0.340	27
765	5	121	72	23	112	26.2	0.245	30
766	1	126	60	0	0	30.1	0.349	47
767	1	93	70	31	0	30.4	0.315	23

Fig 1. The actual feature variables values before data standardization.

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age
0	0.639947	0.848324	0.149641	0.907270	-0.692891	0.204013	0.468492	1.425995
1	-0.844885	-1.123396	-0.160546	0.530902	-0.692891	-0.684422	-0.365061	-0.190672
2	1.233880	1.943724	-0.263941	-1.288212	-0.692891	-1.103255	0.604397	-0.105584
3	-0.844885	-0.998208	-0.160546	0.154533	0.123302	-0.494043	-0.920763	-1.041549
4	-1.141852	0.504055	-1.504687	0.907270	0.765836	1.409746	5.484909	-0.020496
...	...	...	...	...	...	...	...	...
763	1.827813	-0.622642	0.356432	1.722735	0.870031	0.115169	-0.908682	2.532136
764	-0.547919	0.034598	0.046245	0.405445	-0.692891	0.610154	-0.398282	-0.531023
765	0.342981	0.003301	0.149641	0.154533	0.279594	-0.735190	-0.685193	-0.275760
766	-0.844885	0.159787	-0.470732	-1.288212	-0.692891	-0.240205	-0.371101	1.170732
767	-0.844885	-0.873019	0.046245	0.656358	-0.692891	-0.202129	-0.473785	-0.871374

Fig 2. The transformed feature variables values after applying of data standardization technique i.e z-score normalization.

[4] Interactive Development Enviornment

I not only visualized data in the raw and prepared forms but also I plotted dataset before and after applying of data standardization method in Fig 3 and Fig 4 respectively.

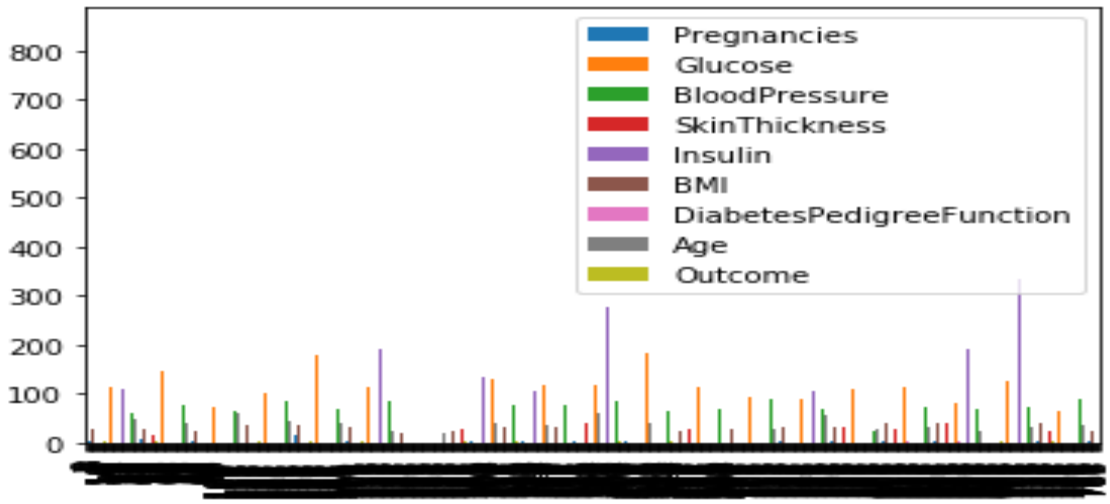

Fig 3. The plotted chart before applying the data standardization method.

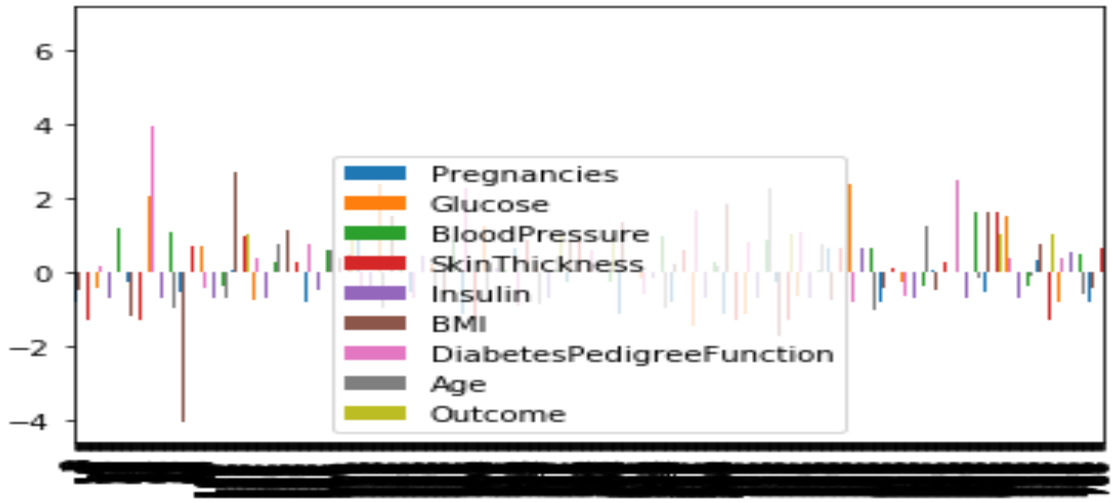

Fig 4. The plotted chart after applying the data standardization method.

By visualizing Fig 3 and Fig 4, It has been noticed that data is in sparsity form in Fig 3. The values of feature variables start from 0 up to around 400. In this situation, it is inefficient to apply any model before data preparation. Due to this reason, firstly I prepared data then go the model implementation phase. See Fig 4, the feature variables values are in standard form as a start from -4 up to 4. In this situation. it is efficient and intuitive to apply the implementation of a model.

After applying data transformation on the dataset, the dataset is partitioned into five different categories and each category is called one round. The partition of the dataset is expressed in the below table.

Table I. Partition of Dataset

	Training	**Testing**
Round # 1	90 %	10 %
Round # 2	80 %	20 %
Round # 3	70 %	30 %
Round # 4	60 %	40 %
Round # 5	50 %	50 %

Models

I have read and explored various models to deal with the above problem. After some research and the current scenario, I have decided on two machine learning models and applied them to the above problem. Then compared the result of each one on different dataset partitions. The models are as follows:

1. Decision Tree
   The decision tree is the predictive model approach used in machine learning, statistics, text mining, data mining, etc. This approach used the decision tree as a prediction model for inspecting items to achieve target items at the leaves. The decision tree is visually express during the decision making processes. The classification technique of decision tree used in the above problem as target value is discrete i.e. 0 or 1.

2. Support Vector Machine (SVM)
   A support vector machine (SVM) is a machine learning classification model and uses to classify a different group of potential data. The SVM model comes in a supervised machine learning approach where data is partitioned into training and testing sets. The training set has features values as well as the target values, while the testing set has only featured values and target values are predicted by the model. Firstly the model is trained for classifying by using training sets. After that trained model takes testing set as input and generates predictions based on model training. It is essential to avoid a subset training set during testing of a model.

Models Visualization

I have implemented SVM and Decision Tree models for this diabetic problem. I have written code in Jupyter Notebook as see in Fig 5 and Fig 6 respectively.

```
In [23]: clf=svm.SVC()       # By using sklearn python library, here I register model as clf.
         clf.fit(newList1,target1) # Here newList1 is training features and target1 is training target variable.
         pprint(clf)  # Display the credential of SVM MODEL

SVC(C=1.0, break_ties=False, cache_size=200, class_weight=None, coef0=0.0,
    decision_function_shape='ovr', degree=3, gamma='scale', kernel='rbf',
    max_iter=-1, probability=False, random_state=None, shrinking=True,
    tol=0.001, verbose=False)
```

Fig 5. The Support Vector Machine model implemented in the python by using the sklearn library.

```
In [40]: train=train_df.iloc[:,:-1]
         target=train_df.iloc[:,-1]
         clf = tree.DecisionTreeClassifier() # By using sklearn python library, here I register model as clf.
         clf = clf.fit(train, target)  # Here newList1 is training features and target1 is training target variable.
         pprint(clf)                   # Display the credential of Decison Tree Predictive Model

DecisionTreeClassifier(ccp_alpha=0.0, class_weight=None, criterion='gini',
                       max_depth=None, max_features=None, max_leaf_nodes=None,
                       min_impurity_decrease=0.0, min_impurity_split=None,
                       min_samples_leaf=1, min_samples_split=2,
                       min_weight_fraction_leaf=0.0, presort='deprecated',
                       random_state=None, splitter='best')
```

Fig 6. The Decision Tree model implemented in the python by using the sklearn library.

After that, I have used the graphviz python library to visualize the decision tree which is responsible for prediction in Fig 7.

```
In [10]: t=["0","1"] # The target values are used in generating graph
         dot_data = tree.export_graphviz(clf, out_file=None,feature_names=df.columns[0:-1],
                                         class_names=t,
                                         filled=True, rounded=True,
                                         special_characters=False)
         graph = graphviz.Source(dot_data) # graphiviz is python-based library which is used for various kinds of visualization


         graph.format = "png"            # Here the image is being saved with png format
         graph.render("Decision Tree")   # The image name is Decision Tree which is being saved in same folder.

Out[10]: 'Decision Tree.png'
```

Fig 7. The Decision Tree is implemented in the python by using the graphviz library.

The graphviz produces the decision tree as see in Fig 8. It is a long tree due to which I have included here the right part of the original decision tree. The complete decision tree lies in the coding folder along with other project files.

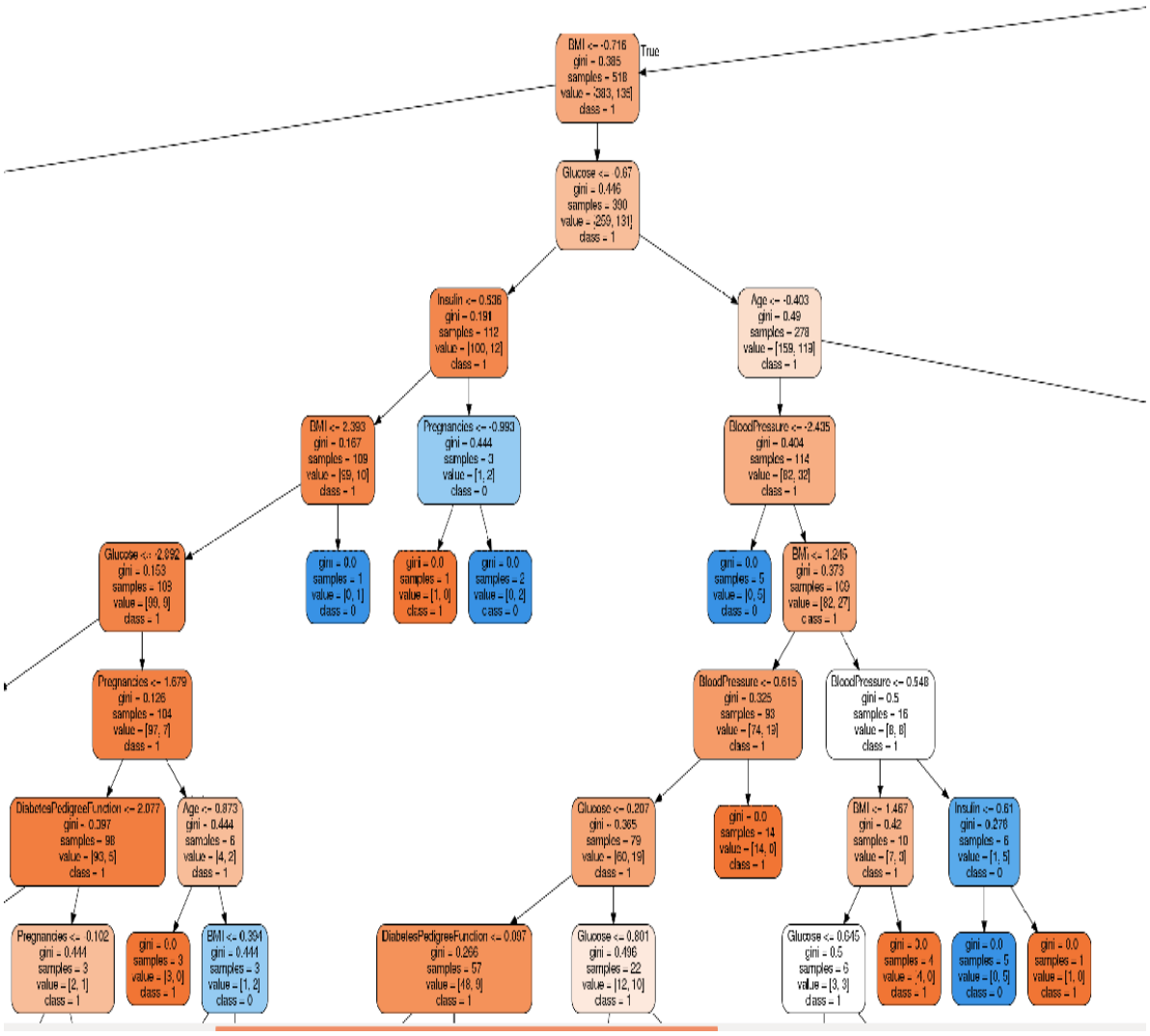

Fig 7. The Decision Tree is generated by using the grapviz python library.

Evaluation

The evaluation of the machine learning model is an integral part of a data science project. The machine learning model is being evaluated to know the performance of the model and how it has been effected for the specific problem. The objective of the evaluation is to compute the accuracy of the trained model on unseen data i.e testing set. There are various evaluation methods are being used in data science projects and I have decided two methods as follows:

1. Multiple Random Sampling

I have applied **multiple random sampling** techniques for the evaluation of two different models. I have gathered results on each round as described in the following tables.

Table II. RESULTS ON THE DIABETES DATASET IN ROUND 1

Models	**Training**	**Testing**	**Accuracy**
Decision Tree	90 %	10 %	77.92207792207793
Support Vector Machine (SVM)	90 %	10 %	83.11688311688312

Table III. RESULTS ON THE DIABETES DATASET IN ROUND 2

Models	**Training**	**Testing**	**Accuracy**
Decision Tree	80 %	20 %	77.27272727272727
Support Vector Machine (SVM)	80 %	20 %	81.16883116883117

Table IV. RESULTS ON THE DIABETES DATASET IN ROUND # 3

Models	**Training**	**Testing**	**Accuracy**
Decision Tree	70 %	30 %	70.86956521739131
Support Vector Machine (SVM)	70 %	30 %	79.56521739130434

Table V. RESULTS ON THE DIABETES DATASET IN ROUND # 4

Models	**Training**	**Testing**	**Accuracy**
Decision Tree	60 %	40 %	69.05537459283387
Support Vector Machine (SVM)	60 %	40 %	79.15309446254072

Table VI. RESULTS ON THE DIABETES DATASET IN ROUND # 5

Models	**Training**	**Testing**	**Accuracy**
Decision Tree	50 %	50 %	68.48958333333334
Support Vector Machine (SVM)	50 %	50 %	77.60416666666666

2. 5- Fold Cross-Validation

I have applied 5-Fold cross-validation techniques i.e 80 % data is used for training purposes and 20 % data is used for testing purposes in the evaluation of two different models as given below.

Table VIII. ACCURACY ON THE DIABETES DATASET BY DECISION TREE

Fold Number	**Accuracy**	**Average Accuracy**
1	70.12987012987013 %	71.0389610389 %
2	63.63636363636364 %	
3	70.77922077922078 %	
4	79.22077922077922 %	
5	71.42857142857143 %	

Table VIII. ACCURACY ON THE DIABETES DATASET BY SUPPORT VECTOR MACHINE

Fold Number	**Accuracy**	**Average Accuracy**
1	76.62337662337663 %	77.0129870129 %
2	70.77922077922078 %	
3	77.92207792207793 %	
4	82.46753246753247 %	
5	77.27272727272727 %	

Conclusion

This is an interesting assignment for me as I get a chance to explore various kinds of the dataset on Kaggle, Github, UCI Machine Learning Repository, and other data science websites. After reading and exploring numerous datasets, I have decided to work on a diabetic dataset. I think data preparation is the most important phase during working on a data science project. I have spent 80 percent time duration on understanding and preparation of data for making reasonable to apply appropriate models. The remaining 20 percent time duration has been spent on the implementation of model and evaluation phases. I have compared two models results in this assignment i.e. Decision Tree and Support Vector Machine (SVM). These models have been applied to a given problem with the help of the Sklean python machine learning library. After evaluating models, I have found that SVM is more efficient for the diabetic dataset problem.

References

- https://www.kaggle.com/avinashbeck/pima-indian-diabetes-dataset/data
- https://github.com/susanli2016/Machine-Learning-with-Python/blob/master/diabetes.csv