17th Mar Assignment

March 20, 2023

1 Assignment 41

Q1: What are missing values in a dataset? Why is it essential to handle missing values? Name some algorithms that are not affected by missing values.

Ans. Missing values in a dataset refer to the absence of data points for one or more variables in a particular observation. There are several reasons why data may be missing, such as data entry errors, data processing issues, or non-response by respondents. Missing data can be categorized into three types: missing completely at random, missing at random, and missing not at random.

Handling missing values is essential because they can negatively impact the accuracy of statistical models and analysis. If left unaddressed, missing data can lead to biased estimates, reduced statistical power, and incorrect conclusions. It is crucial to deal with missing data to ensure the validity and reliability of data analysis results.

Some algorithms that are not affected by missing values include decision trees, random forests, and support vector machines. These algorithms can handle missing values by imputing missing data or using other techniques that do not require complete data. However, it is still important to deal with missing values before applying any statistical model to ensure optimal performance.

Q2: List down techniques used to handle missing data. Give an example of each with python code.

Ans.There are several techniques to handle missing data, including:

1. Deletion: This technique involves removing observations or variables with missing values. There are three types of deletion methods: listwise deletion, pairwise deletion, and casewise deletion.

[2]:
```python
import pandas as pd

# create a sample dataframe with missing values
df = pd.DataFrame({'A': [1, 2, 3, 4, None], 'B': [5, None, 7, 8, 9]})
print(df)
```

```
     A    B
0  1.0  5.0
1  2.0  NaN
2  3.0  7.0
3  4.0  8.0
4  NaN  9.0
```

[3]:
```python
# using listwise deletion
df1 = df.dropna()
print(df1)
```

```
     A    B
0  1.0  5.0
2  3.0  7.0
3  4.0  8.0
```

[4]:
```python
# using pairwise deletion
df2 = df.dropna(axis=1, how='any')
print(df2)
```

```
Empty DataFrame
Columns: []
Index: [0, 1, 2, 3, 4]
```

[6]:
```python
# using casewise deletion
# look missing values in column A and delete that row
df3 = df.dropna(subset=['A'])
print(df3)
```

```
     A    B
0  1.0  5.0
1  2.0  NaN
2  3.0  7.0
3  4.0  8.0
```

2. Imputation: This technique involves replacing missing values with estimated values based on other non-missing values.

[7]:
```python
import pandas as pd
from sklearn.impute import SimpleImputer

# create a sample dataframe with missing values
df = pd.DataFrame({'A': [1, 2, 3, None, 5], 'B': [5, 6, None, 8, 9]})
```

[8]:
```python
df
```

[8]:
```
     A    B
0  1.0  5.0
```

```
1  2.0  6.0
2  3.0  NaN
3  NaN  8.0
4  5.0  9.0
```

[9]:
```python
# using mean imputation
imputer = SimpleImputer(strategy='mean')
df1 = pd.DataFrame(imputer.fit_transform(df), columns=df.columns)
print(df1)
```

```
      A    B
0  1.00  5.0
1  2.00  6.0
2  3.00  7.0
3  2.75  8.0
4  5.00  9.0
```

[10]:
```python
# using median imputation
imputer = SimpleImputer(strategy='median')
df2 = pd.DataFrame(imputer.fit_transform(df), columns=df.columns)
print(df2)
```

```
     A    B
0  1.0  5.0
1  2.0  6.0
2  3.0  7.0
3  2.5  8.0
4  5.0  9.0
```

[11]:
```python
# using mode imputation
imputer = SimpleImputer(strategy='most_frequent')
df3 = pd.DataFrame(imputer.fit_transform(df), columns=df.columns)
print(df3)
```

```
     A    B
0  1.0  5.0
1  2.0  6.0
2  3.0  5.0
3  1.0  8.0
4  5.0  9.0
```

Q3: Explain the imbalanced data. What will happen if imbalanced data is not handled?

Ans. Imbalanced data refers to a situation where the classes or categories in a dataset are not represented equally, and one class has significantly fewer samples than the other(s). For example, in a binary classification problem, if 95% of the samples belong to class A and only 5% to class B, then the dataset is imbalanced.

If imbalanced data is not handled, it can lead to several problems. One major issue is that the predictive model will be biased towards the majority class, resulting in poor performance on the minority class. This is because the model will be trained to optimize the overall accuracy of the dataset, which will be high due to the high number of samples in the majority class. As a result, the model may fail to identify patterns in the minority class, leading to poor classification performance.

Another problem is that the evaluation metrics used to assess model performance may not accurately reflect its true performance. For example, using accuracy as the evaluation metric can be misleading in an imbalanced dataset since the model may achieve high accuracy by simply predicting the majority class for all samples.

Handling imbalanced data is essential to improve the performance of predictive models and obtain accurate results. Some techniques used to handle imbalanced data include resampling, using different evaluation metrics, using cost-sensitive learning, and using ensemble methods.

Q4: What are Up-sampling and Down-sampling? Explain with an example when up-sampling and down-sampling are required.

Ans. Upsampling and downsampling are techniques used to handle imbalanced data by adjusting the number of samples in each class.

- Upsampling involves increasing the number of samples in the minority class to balance it with the majority class. This can be done by either replicating existing samples or generating new synthetic samples using techniques such as SMOTE (Synthetic Minority Over-sampling Technique). Example: Suppose we have a dataset with two classes, A and B, where class A has 1000 samples and class B has only 100 samples. This is an imbalanced dataset, and we want to balance it using upsampling. We can replicate the existing samples in class B to increase their number to 1000, making the dataset balanced.
- Downsampling involves decreasing the number of samples in the majority class to balance it with the minority class. This can be done by randomly selecting a subset of samples from the majority class. Example: Suppose we have a dataset with two classes, A and B, where class A has 1000 samples and class B has only 100 samples. This is an imbalanced dataset, and we want to balance it using downsampling. We can randomly select 100 samples from class A to match the number of samples in class B, making the dataset balanced.

Q5: What is data Augmentation? Explain SMOTE.

Ans. Data augmentation is a technique used in machine learning to generate additional training data from the existing data by applying various transformations such as rotations, scaling, flips, and other random perturbations. Data augmentation is used to increase the size of the dataset, reduce overfitting, and improve the performance of the model.

SMOTE (Synthetic Minority Over-sampling Technique) is a data augmentation technique that is commonly used to handle imbalanced datasets. SMOTE works by generating synthetic samples in the minority class by interpolating between existing samples.

Here's how SMOTE works:

1. For each sample in the minority class, SMOTE selects k nearest neighbors from the minority class, where k is a user-defined parameter.
2. SMOTE then generates new synthetic samples by interpolating between the original sample and its k nearest neighbors.
3. The synthetic samples are generated by randomly selecting a point along the line segment joining the original sample and one of its k nearest neighbors. #### SMOTE is useful when we have an imbalanced dataset, and we want to generate additional synthetic samples in the minority class to balance it with the majority class. By doing so, we can train a model that can learn from both classes effectively and make accurate predictions.

Here's an example of how to use SMOTE in Python:

[13]:
```
pip install imblearn
```

```
Collecting imblearn
  Downloading imblearn-0.0-py2.py3-none-any.whl (1.9 kB)
Collecting imbalanced-learn
  Downloading imbalanced_learn-0.10.1-py3-none-any.whl (226 kB)
                        226.0/226.0 kB
10.2 MB/s eta 0:00:00
Requirement already satisfied: joblib>=1.1.1 in
/opt/conda/lib/python3.10/site-packages (from imbalanced-learn->imblearn)
(1.2.0)
Requirement already satisfied: scipy>=1.3.2 in /opt/conda/lib/python3.10/site-
packages (from imbalanced-learn->imblearn) (1.9.3)
Requirement already satisfied: scikit-learn>=1.0.2 in
/opt/conda/lib/python3.10/site-packages (from imbalanced-learn->imblearn)
(1.2.0)
Requirement already satisfied: threadpoolctl>=2.0.0 in
/opt/conda/lib/python3.10/site-packages (from imbalanced-learn->imblearn)
(3.1.0)
Requirement already satisfied: numpy>=1.17.3 in /opt/conda/lib/python3.10/site-
packages (from imbalanced-learn->imblearn) (1.23.5)
Installing collected packages: imbalanced-learn, imblearn
Successfully installed imbalanced-learn-0.10.1 imblearn-0.0
Note: you may need to restart the kernel to use updated packages.
```

[14]:
```
from imblearn.over_sampling import SMOTE
from sklearn.datasets import make_classification
from collections import Counter
```

```
# Generate an imbalanced dataset
X, y = make_classification(n_classes=2, class_sep=2,
                           weights=[0.1, 0.9], n_informative=3,
                           n_redundant=1, flip_y=0, n_features=20,
                           n_clusters_per_class=1, n_samples=1000,␣
 ↪random_state=10)

# Print the class distribution
print('Original dataset shape:', Counter(y))

# Create an instance of SMOTE
smote = SMOTE(random_state=42)

# Apply SMOTE to the dataset
X_resampled, y_resampled = smote.fit_resample(X, y)

# Print the class distribution after SMOTE
print('Resampled dataset shape:', Counter(y_resampled))
```

```
Original dataset shape: Counter({1: 900, 0: 100})
Resampled dataset shape: Counter({0: 900, 1: 900})
```

Q6: What are outliers in a dataset? Why is it essential to handle outliers?

Ans. Outliers in a dataset are data points that deviate significantly from the other observations in the dataset. Outliers can occur due to measurement errors, data entry errors, or because they represent genuine observations that are rare or extreme. Outliers can be identified using statistical methods, such as box plots, scatter plots, or z-scores.

It is essential to handle outliers because they can have a significant impact on the results of data analysis or machine learning algorithms. Outliers can affect the mean and standard deviation of the data, making it difficult to estimate the central tendency and variability of the dataset. Outliers can also influence the parameters of a model and lead to biased predictions.

Handling outliers can involve several strategies, including removing the outliers, transforming the data to reduce the influence of the outliers, or using robust statistical methods that are less sensitive to outliers. The specific approach to handling outliers depends on the nature of the data and the analysis or modeling goals.

In summary, handling outliers is essential because they can significantly impact the results of data analysis and machine learning algorithms. By identifying and handling outliers appropriately, we can improve the accuracy and robustness of our models and ensure that the data is suitable for the intended analysis or application.

Q7: You are working on a project that requires analyzing customer data. However, you notice that some of the data is missing. What are some techniques you can use to handle the missing data in your analysis?

Ans. There are several techniques that can be used to handle missing data in customer data analysis. Here are some of the most common techniques:

1. Deletion: One approach to handling missing data is to simply delete the rows or columns that contain missing values. This approach is simple and straightforward, but it can result in a loss of data and may bias the results if the missing values are not randomly distributed.
2. Imputation: Imputation involves estimating the missing values based on the observed values in the dataset. There are several methods for imputation, including mean imputation, median imputation, and regression imputation.
3. Multiple Imputation: Multiple imputation is a technique that involves creating several plausible imputed datasets, each with its own set of imputed values. This approach accounts for the uncertainty in the imputed values and can improve the accuracy of the analysis.
4. Using Machine Learning algorithms: Another approach to handling missing data is to use machine learning algorithms that are robust to missing values, such as decision trees, random forests, and gradient boosting. These algorithms can handle missing values by imputing the missing values or ignoring them during model training.
5. Domain Expert: Finally, domain experts may be able to provide insight or guidance on how to handle missing data based on their knowledge of the data and the context of the analysis.

The choice of which technique to use depends on the nature of the data, the amount of missing data, and the goals of the analysis. It's important to carefully consider the pros and cons of each approach before selecting the best technique for handling the missing data.

Q8: You are working with a large dataset and find that a small percentage of the data is missing. What are some strategies you can use to determine if the missing data is missing at random or if there is a pattern to the missing data?

Ans. When working with a large dataset and missing data, it's important to determine if the missing data is missing at random (MAR) or if there is a pattern to the missing data. Here are some strategies to determine if the missing data is missing at random:

1. Visualize the data: One way to determine if the missing data is missing at random is to visualize the data using plots such as histograms, box plots, or scatter plots. This can help identify any patterns or trends in the data.
2. Use statistical tests: Statistical tests such as t-tests, ANOVA, or correlation tests can be used to test if the missing data is missing at random. If there is no significant difference between the missing and non-missing data, then it may be assumed that the missing data is missing at random.
3. Imputation: Imputation is a technique used to fill in missing values with estimated values. If there is a pattern to the missing data, then imputation can be used to estimate missing

values. If the imputed values are significantly different from the non-missing data, then it may be assumed that the missing data is not missing at random.

4. Domain Knowledge: Having domain knowledge can also help to determine if the missing data is missing at random or not. If there is a reason for the missing data, such as a technical problem, then it may be assumed that the missing data is not missing at random.

It's important to note that determining if the missing data is missing at random can be a challenging task and there may be no definitive answer. However, using these strategies can help to identify any patterns or trends in the missing data and provide insight into the nature of the missing data.

Q9: Suppose you are working on a medical diagnosis project and find that the majority of patients in the dataset do not have the condition of interest, while a small percentage do. What are some strategies you can use to evaluate the performance of your machine learning model on this imbalanced dataset?

Ans. When dealing with imbalanced datasets, such as in a medical diagnosis project where the majority of patients do not have the condition of interest, there are several strategies that can be used to evaluate the performance of a machine learning model. Here are some of the most commonly used strategies:

1. Confusion Matrix: The confusion matrix is a table that displays the true positive, false positive, true negative, and false negative rates of a classifier. This can help to evaluate the performance of the model on the minority class.
2. Precision-Recall Curve: Precision-Recall (PR) curve is a graphical representation of the precision and recall rates of a classifier. This can help to evaluate the model's performance on the minority class and compare different models.
3. ROC Curve: Receiver Operating Characteristic (ROC) curve is a graphical representation of the true positive rate (sensitivity) against the false positive rate (1-specificity) of a classifier. This can help to evaluate the model's overall performance on both the majority and minority classes.
4. F1 Score: The F1 score is the harmonic mean of precision and recall. This metric can help to evaluate the model's performance on the minority class.
5. Cost-Sensitive Learning: Cost-sensitive learning involves modifying the loss function of the model to account for the imbalanced nature of the dataset. This can help to improve the model's performance on the minority class.
6. Sampling Techniques: Sampling techniques such as oversampling the minority class or undersampling the majority class can help to balance the dataset and improve the performance of the model on the minority class.

It's important to note that selecting the appropriate evaluation strategy depends on the specific goals of the project and the characteristics of the dataset. A combination of these strategies may be required to adequately evaluate the performance of a machine learning model on an imbalanced dataset.

Q10: When attempting to estimate customer satisfaction for a project, you discover that the dataset is unbalanced, with the bulk of customers reporting being satisfied. What methods can you employ to balance the dataset and down-sample the majority class?

Ans. When dealing with an imbalanced dataset in which the majority of customers report being satisfied, there are several methods that can be used to balance the dataset and down-sample the majority class. Here are some of the most commonly used methods:

1. Undersampling: This involves randomly selecting a subset of the majority class to create a more balanced dataset. However, this approach can result in the loss of important information and may not be suitable for small datasets.
2. Oversampling: This involves increasing the number of instances in the minority class by generating synthetic data points using techniques such as SMOTE. This approach can improve the performance of the model on the minority class, but may also result in overfitting.
3. Cost-sensitive learning: This involves modifying the loss function of the model to account for the imbalance in the dataset. This approach can be used to assign different weights to the different classes to balance the dataset.
4. Ensemble methods: Ensemble methods such as bagging and boosting can be used to combine multiple models trained on different subsets of the data to create a more robust and accurate model.
5. Stratified Sampling: This involves splitting the data into balanced subsets based on the target variable. For example, if there are two classes, satisfied and dissatisfied customers, the data can be split into two equal-sized subsets, each containing an equal proportion of satisfied and dissatisfied customers.
6. Hybrid Sampling: This involves using a combination of over and undersampling techniques to create a more balanced dataset.

It's important to note that selecting the appropriate method depends on the specific goals of the project and the characteristics of the dataset. A combination of these methods may be required to effectively balance the dataset and down-sample the majority class.

Q11: You discover that the dataset is unbalanced with a low percentage of occurrences while working on a project that requires you to estimate the occurrence of a rare event. What methods can you employ to balance the dataset and up-sample the minority class?

Ans. When dealing with an imbalanced dataset in which the minority class is rare, there are several methods that can be used to balance the dataset and up-sample the minority class. Here are some commonly used methods:

1. Oversampling: This involves increasing the number of instances in the minority class by generating synthetic data points using techniques such as SMOTE or ADASYN. This approach

can improve the performance of the model on the minority class, but may also result in overfitting.

2. Cost-sensitive learning: This involves modifying the loss function of the model to account for the imbalance in the dataset. This approach can be used to assign different weights to the different classes to balance the dataset.
3. Ensemble methods: Ensemble methods such as bagging and boosting can be used to combine multiple models trained on different subsets of the data to create a more robust and accurate model.
4. Stratified Sampling: This involves splitting the data into balanced subsets based on the target variable. For example, if there are two classes, rare and non-rare occurrences, the data can be split into two equal-sized subsets, each containing an equal proportion of rare and non-rare occurrences.
5. Hybrid Sampling: This involves using a combination of over and undersampling techniques to create a more balanced dataset.

It's important to note that selecting the appropriate method depends on the specific goals of the project and the characteristics of the dataset. A combination of these methods may be required to effectively balance the dataset and up-sample the minority class. It's also important to evaluate the performance of the model on both the minority and majority class to ensure that the model is not biased towards one class.

[ ]:

```

```