Príprava a predspracovanie dát

Normalizácia pomocou Min Max Scaleru:

```
from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler()
data['atribut'] = pd.DataFrame(scaler.fit_transform(pd.DataFrame(data['atribut']. )), columns=['fare'])
```

Transformácia kategorického atribútu na numerický pomocou Label Encoder:

```
from sklearn.preprocessing import LabelEncoder
data['atribut'] = LabelEncoder().fit_transform(data['atribut'])
```

Transformácia ordinálnych kategorických atribútov na numerické:

```
data['atribut'] = data['atribut'].map({Hodnota1: 0, Hodnota2: 1, Hodnota3: 2})
```

Binarizácia kategorického atribútu (One Hot Encoding):

```
data = pd.get_dummies(data, columns=[atribut'])
```

Rozdelenie na trénovaciu a testovaciu množinu

```
X_data = data.drop("cieľový atribút", axis=1)
y_data = data["cieľový atribút"]

from sklearn.model_selection import train_test_split
 X_train, X_test, y_train, y_test = train_test_split(X_data, y_data, test_size=0.3, random_state=123)
```

Vyhodnotenie modelov

Klasifikácia

```
y_model = model.predict(X_test)

from sklearn.metrics import accuracy_score, precision_score,
recall_score
accuracy_score(y_test, y_model)
precision_score(y_test, y_model)
recall_score(y_test, y_model)

from sklearn.metrics import confusion_matrix
print(confusion_matrix(y_test, y_model)
```

Regresia

```
y_model = model.predict(X_test)

summary_df = pd.DataFrame()
summary_df['target'] = y_test
summary_df['prediction'] = y_model
print(summary_df)

from sklearn.metrics import mean_absolute_error, mean_squared_error,
r2_score

mae = mean_absolute_error(y_test, y_model)
mse = mean_squared_error(y_test, y_model)
r2 = r2_score(y_test, y_model)
```

Metriky pre zhlukovanie

```
labels = model.predict(X_train)

print(model.inertia_)
print(euclidean_distances(model.cluster_centers_))
print(model.labels_)

cluster_0 = np.where(model.labels_==0)
data_cluster_0.describe()
```

```
from sklearn.metrics import silhouette_score

labels = model.predict(X_train)
print(silhouette_score(X_train, labels))
```

Modelovanie

Prediktívne

Klasifikácia

K-NN:

```
from sklearn.neighbors import KNeighborsClassifier
model = KNeighborsClassifier()
model.fit(X_train, y_train)
```

Stromy:

```
from sklearn.tree import DecisionTreeClassifier
model = DecisionTreeClassifier()
model.fit(X_train, y_train)
```

Random Forests:

```
from sklearn.ensemble import RandomForestClassifier
model = RandomForestClassifier()
model.fit(X, Y)
```

Naive Bayes:

```
from sklearn.naive_bayes import GaussianNB
model = GaussianNB()
model.fit(X_train, y_train)
```

Support Vector Machines

```
from sklearn.svm import SVC
model = SVC()
model.fit(X_train, y_train)
```

Regresia

Lineárna regresia:

```
from sklearn.linear_model import LinearRegression
model = LinearRegression()
model.fit(X_train, y_train)
```

K-NN :

```
from sklearn.neighbors import KNeighborsRegressor
model = KNeighborsRegressor()
model.fit(X_train, y_train)
```

Stromy:

```
from sklearn.tree import DecisionTreeRegresssor
model = DecisionTreeRegressor()
model.fit(X_train, y_train)
```

Popisné

K-Means

```
from sklearn.cluster import KMeans
model = KMeans(n_clusters=4)
model.fit(X_train)
```

```
from sklearn.cluster import DBSCAN

dbscan = DBSCAN(eps=100)
labels = dbscan.fit_predict(X_train)
```

Ladenie parametrov modelu

Nastavenie rozsahu parametrov

```
parameter_range = list(range(1, 50))
param_grid = dict(<názov parametra modelu>=parameter_range)
```

Použitie Grid Search s krížovou validáciou pre odhad kvality modelu

```
from sklearn.model_selection import GridSearchCV
grid = GridSearchCV(estimator=model, param_grid=param_grid, cv=10, scoring='accuracy')
grid.fit(X_train, y_train)
```

Nájdenie najlepšieho modelu

```
print(grid.best_params_)
print(grid.best_score_)
```